**UNIVERSIDADE FEDERAL DE SERGIPE**

**CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS**

**PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO**

# O TRABALHO DO ARQUITETO DE SOFTWARE NO BRASIL: PADRÕES E ANTIPADRÕES

**Manoela dos Reis Oliveira**

**São Cristóvão**

**2019**

**UNIVERSIDADE FEDERAL DE SERGIPE**

**CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS**

**PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO**

**Manoela dos Reis Oliveira**

# O TRABALHO DO ARQUITETO DE SOFTWARE NO BRASIL: PADRÕES E ANTIPADRÕES

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação (PROCC) da Universidade Federal de Sergipe (UFS) como parte de requisito para obtenção do título de Mestre em Ciência da Computação.

Orientador: Prof. Dr. Michel dos Santos Soares

São Cristóvão
2019

**FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA CENTRAL**
**UNIVERSIDADE FEDERAL DE SERGIPE**

Oliveira, Manoela dos Reis
O48t O trabalho do arquiteto de software no Brasil : padrões e antipadrões / Manoela dos Reis Oliveira ; orientador Michel dos Santos Soares. - São Cristóvão, 2020.
89 f. : il.

Dissertação (mestrado em Ciência da Computação) – Universidade Federal de Sergipe, 2020.

1. Computação. 2. Arquitetura de software. 3. Pessoal da área de processamento eletrônico de dados – Pesquisa. 4. Levantamentos. I. Soares, Michel dos Santos, orient. II. Título.

CDU 004.4

**Manoela dos Reis Oliveira**

# O TRABALHO DO ARQUITETO DE SOFTWARE NO BRASIL: PADRÕES E ANTIPADRÕES

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação (PROCC) da Universidade Federal de Sergipe (UFS) como parte de requisito para obtenção do título de Mestre em Ciência da Computação.

## BANCA EXAMINADORA

Prof. Dr. Michel dos Santos Soares, Presidente
Universidade Federal de Sergipe (UFS)

Prof. Dr. Rogério Patrício Chagas do Nascimento
Universidade Federal de Sergipe (UFS)

Profª. Drª. Adicinéia Aparecida de Oliveira
Universidade Federal de Sergipe (UFS)

Profª. Drª. Joyce Meire da Silva França
Instituto Federal de Educação Ciência e Tecnologia do Norte de Minas Gerais (IFNMG)

**UNIVERSIDADE FEDERAL DE SERGIPE**
**PRÓ-REITORIA DE PÓS-GRADUAÇÃO E PESQUISA**
**COORDENAÇÃO DE PÓS-GRADUAÇÃO**
**PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO**

**Ata da Sessão Solene de Defesa da Dissertação do Curso de Mestrado em Ciência da Computação-UFS.**
**Candidato:** MANOELA DOS REIS OLIVEIRA

Em 26 dias do mês de novembro do ano de dois mil e dezenove, com início às 14h30min, realizou-se na Sala de Seminário do DCOMP da Universidade Federal de Sergipe, na Cidade Universitária Prof. José Aloísio de Campos, a Sessão Pública de Defesa de Dissertação de Mestrado do candidato **Manoela dos Reis Oliveira,** que desenvolveu o trabalho intitulado: **" *O TRABALHO DO ARQUITETO DE SOFTWARE NO BRASIL: PADRÕES E ANTIPADRÕES*",** sob a orientação do Prof. Dr. **Michel dos Santos Soares**. A Sessão foi presidida pelo Prof. Dr. **Michel dos Santos Soares** (PROCC/UFS), que após a apresentação da dissertação passou a palavra aos outros membros da Banca Examinadora, Prof. Dr. **Rogério Patrício Chagas do Nascimento** (PROCC/UFS), a Profª **Adicinéia Aparecida de Oliveira** (Dcomp/UFS) e, em seguida, ao Profª. **Joyce Meire da Silva França** (IFNMG) . Após as discussões, a Banca Examinadora reuniu-se e considerou o mestrando (a ) Aprovada *"(aprovado/reprovado)"*. Atendidas as exigências da Instrução Normativa 01/2017/PROCC, do Regimento Interno do PROCC (Resolução 67/2014/CONEPE), e da Resolução nº 25/2014/CONEPE que regulamentam a Apresentação e Defesa de Dissertação, e nada mais havendo a tratar, a Banca Examinadora elaborou esta Ata que será assinada pelos seus membros e pelo mestrando.

Cidade Universitária *"Prof. José Aloísio de Campos"*, 26 de novembro de 2019.

Michel S. Soares
**Prof. Dr. Michel dos Santos Soares**
**(PROCC/UFS)**
**Presidente**

**Prof. Dr. Rogério Patrício Chagas do Nascimento**
**(PROCC/UFS)**
**Examinador Interno**

**Prof. Dr. Adicinéia Aparecida de Oliveira**
**(Dcomp/UFS)**
**Examinador Interno**

Joyce França
**Profª. Drª. Joyce Meire da Silva França**
**(IFNMG)**
**Examinador Externo**

Manoela dos Reis Oliveira
**Manoela dos Reis Oliveira**
**Candidato**

*Nós só podemos ver um pouco do futuro,*
*mas o suficiente para perceber que há muito a fazer.*
*(Alan Turing)*

# Agradecimentos

A Deus, por ter me dado saúde e forças para superar as dificuldades e sabedoria necessária a esse momento da minha vida.

Aos meus pais, Manoel dos Reis Oliveira Filho e Maria Edna Santos Brito Oliveira, que sempre me deram toda condição e apoio para que alcançasse meus objetivos por meio do estudo, oferecendo muito amor e momentos de paz durante esse árduo caminho. Aos meus avós Raimunda e Manoel *(in memorian)* que estão felizes por mim mesmo que em outro plano.

Ao meu amor, Felipe Vieira, por ter se tornado o companheiro perfeito nessa jornada chamada vida, decidiu segurar minha mão e seguir ao meu lado para que juntos vencêssemos essa batalha. Foram finais de semana de estudo em conjunto, horas e horas conversando sobre o mestrado onde um foi o ponto de apoio inabalável do outro. Você é o melhor presente que o mestrado poderia ter me dado!

Ao meu orientador, Michel, pela paciência e dedicação, sempre me guiando pelos caminhos do mestrado e acreditando no meu potencial quando eu mesma não acreditei, foram inúmeras as vezes que proferi não merecer o orientador que tenho. Exprimo aqui o meu enorme agradecimento!

Aos professores Drª. Adicinéia Aparecida de Oliveira, Dr. Rogério Patrício Nascimento Chagas e Drª Joyce Meire da Silva França, meus sinceros agradecimentos por terem aceitado o convite para participar de minha Banca.

Aos meus amigos, Katarina, Larissa e Gabriel, por dividirem comigo tantos anos de vida e terem vibrado de felicidade por cada uma das minhas vitórias. A Jiselle, Jéssica e Pablo por terem sido ombro amigo e com paciência e carinho ouviram todas as minhas lamúrias e me incentivaram a continuar. A Fernanda, Daiana e Anne por sempre me encorajarem e nunca desistirem de mim mesmo depois de todas as negativas para sair por precisar estudar. Aos meus amigos do vale (Ismael, Samir, Bel e Nilson) e os Eternos (Paty, Dodô, Lipe, Jorginho, Jose e Ib), obrigada por cada um da sua forma se fazer presente, torcer por mim e me apoiar. Todos vocês me fazem uma pessoa melhor!

Aos meus colegas de mestrado, Flaygner, Luís, Ademir e Leonardo, pelos momentos de motivação e descontração.

Aos meus amigos e ex-colegas de trabalho da Stefanini, podemos ter seguido caminhos diferentes mas os momentos que vivemos nunca serão esquecidos.

Finalmente, sou grata a todas as pessoas que contribuíram direta ou indiretamente pra chegar onde estou hoje.

A todos vocês, meu muito obrigado!

# Resumo

Embora as habilidades e os conhecimentos dos arquitetos de software já tenham sido objeto de estudos nos últimos anos, os pesquisadores e profissionais ainda não chegaram a um consenso claro sobre as atividades que um arquiteto de software é responsável na prática, a fim de ser bem-sucedido na profissão. Nos últimos anos, devido à ocorrência de sucessivas mudanças e evolução de novas tecnologias, os papéis do arquiteto e até mesmo as práticas relacionadas à arquitetura de software foram continuamente alterados no ciclo de vida de desenvolvimento de software. É esperado que o arquiteto de software possua uma diversidade de habilidades. Além do conhecimento técnico, conhecimento de domínio e as habilidades de comunicação devem ser considerados. No entanto, há muitas ofertas de emprego para essa posição que têm habilidades desejadas e funções a serem desempenhadas totalmente diferentes das já conhecidas e consideradas essenciais pelos estudos acadêmicos e industriais. Com o objetivo de entender melhor o que os arquitetos de software realmente fazem em suas atividades cotidianas na prática, e em como isso se assemelha ou se distancia das habilidades, papéis e conhecimentos citados na literatura como essenciais, neste trabalho foi conduzida uma pesquisa em larga escala com 536 profissionais atuando no Brasil que atualmente trabalham ou trabalharam em algum período de suas carreiras como Arquitetos de Software. Entre os resultados, está claro que as funções, responsabilidades, atividades e tarefas desempenhadas pelos arquitetos de software ainda são amplamente desconhecidas e difusas nas organizações, uma vez que tarefas importantes a serem executadas pelos arquitetos de software ainda não são senso comum na indústria.

**Palavras-chaves**: Arquitetura de Software, Survey em larga escala, Arquiteto de Software.

# Abstract

Although the skills and knowledge of software architects have already been the subject of some studies in recent years, researchers and practitioners still have not come to a clear consensus about the activities that a software architect is often responsible in practice in order to be considered successful. In recent years, due to occurrence of successive changes and evolution of new technologies, the architect´roles and even practices related to software architecture have been continuously changed in the software development life cycle. It is expected that a software architect possess a diversity of skills. In addition to technical knowledge, domain knowledge and communication skills must be considered. However, there are many job offerings for this position which have skills and roles totally different from the ones already known and considered essential by academic and industry studies. In order to better understand what software architects actually do in their daily activities in practice, and how this resembles or distances themselves from the skills, roles and knowledge cited in the literature as essential, in this work we have conducted a large-scale survey with 536 professionals working in Brazil who currently work or have worked at some period in their careers as software architects. Among the results, it is clear that the roles, responsibilities, activities and tasks performed by software architects are still largely unknown and diffuse in organizations, as important tasks to be performed by software architects are still not common sense in industry.

**Key-words**: Software Architecture, Large-scale Survey, Software Architect.

# Lista de figuras

# Lista de quadros

# Lista de tabelas

# Lista de abreviaturas e siglas

| | |
|---|---|
| CC | Ciência da Computação |
| DIN-UEM | Departamento de Informática da Universidade Estadual de Maringá |
| EC | Engenharia da Computação |
| ES | Engenharia de Software |
| IES | Instituições de Ensino Superior |
| IFGoiano | Instituto Federal de Educação, Ciência e Tecnologia Goiano |
| IFNMG | Instituto Federal de Educação, Ciência e Tecnologia do Norte de Minas Gerais |
| OTAN | Organização do Tratado do Atlântico Norte |
| SI | Sistemas de Informação |
| TI | Tecnologia da Informação |
| UDESC | Fundação Universidade do Estado de Santa Catarina |
| UEG | Universidade Estadual de Goiás |
| UEM | Universidade Estadual de Maringá |
| UFC | Universidade Federal do Ceará |
| UFERSA | Universidade Federal Rural do Semi-Árido |
| UFMS | Universidade Federal do Mato Grosso do Sul |
| UFRN | Universidade Federal do Rio Grande do Norte |
| UFS | Universidade Federal de Sergipe |
| UNB | Universidade de Brasília |
| USP | Universidade de São Paulo |
| UTFPR | Universidade Tecnológica Federal do Paraná |

# Sumário

# 1 Introdução

Este Capítulo descreve as principais motivações que levaram ao desenvolvimento deste trabalho, no qual serão listados a contextualização da pesquisa, objetivos, metodologia e trabalhos relacionados a partir dos quais é possível obter conhecimento geral sobre o contexto de aplicação deste trabalho.

## 1.1 Contextualização

Até o final da década de 1980, a palavra "arquitetura" foi usada principalmente no sentido de descrever a arquitetura do sistema, ou seja, a estrutura física de um sistema de computador ou, às vezes, no sentido mais restrito de um determinado conjunto de instruções da família de computadores.

A palavra arquitetura também foi frequentemente utilizada como referência à estrutura dos sistemas de software em larga escala ou, com menos frequência, no sentido de um conjunto de instruções computacionais (SHAW; CLEMENTS, 2006).

Nas últimas décadas, os papéis da arquitetura de software e do arquiteto de software dentro do ciclo de vida do desenvolvimento de software evoluíram bastante (SHERMAN; HADAR, 2015). Sherman e Hadar afirmam que à medida que novas metodologias de desenvolvimento emergiram e começaram e ser usadas amplamente, o escopo e definição de diferentes papéis nas equipes de desenvolvimento mudou.

A ascensão dos sistemas de informação em larga escala evidenciou a importância da arquitetura de software. Sucessivas mudanças e evoluções nos últimos anos culminaram na necessidade de arquitetos de software que atuassem como mentores e elementos centrais entre diferentes projetos, equipes, domínios e camadas da organização. Devido a esta evolução surgiu a necessidade de ampliar o escopo do papel do arquiteto de software além das funções tradicionais, como documentar o sistema e realizar definições iniciais, adicionando uma importância crítica no papel do arquiteto como elemento de ligação (HOHPE et al., 2016).

É necessário ter em mente que é exigido do arquiteto de software um equilíbrio entre o conhecimento técnico, conhecimento de domínio e habilidades de comunicação (KRUCHTEN, 2008), (CORREA, 2013). Em contrapartida, existem ofertas de emprego para esse cargo que possuem em sua descrição funções e atribuições totalmente diferentes das funções consideradas essenciais no desempenho do trabalho de um arquiteto de software. Em (GORTON, 2011), o autor enfatiza que analisando as ofertas de trabalhos é possível encontrar variados cargos de arquiteto, como por exemplo: arquitetos de produtos, arquitetos técnicos, arquitetos de soluções, arquitetos empresariais, entre outros. Torna-se difícil distinguir quais competências diferenciam

de fato um cargo do outro.

Em (ANTONINO; MORGENSTERN; KUHN, 2016), os autores observaram que clientes atribuíram a função de arquiteto de software a pessoas que não se encaixavam no perfil correspondente. Em entrevistas espontâneas com arquitetos de software embarcados, eles descobriram que a maioria desses profissionais conseguiram seu papel porque trabalhavam para a empresa há anos e eram engenheiros com habilidades técnicas reconhecidas na organização. Muitos deles se encaixam no perfil de um engenheiro sênior, não de um arquiteto de software.

Em (CLEMENTS et al., 2007), os deveres, competências e conhecimentos dos arquitetos de software são investigados. Como resultado, os autores concluíram que o trabalho dos arquitetos de software é muito mais complexo do que apenas tomar decisões técnicas, embora a tomada de decisões claramente continue sendo uma atividade essencial ao cargo. Os autores também detectaram a necessidade de uma visão abrangente e definição formal sobre o que é necessário para que um profissional seja considerado um arquiteto de software verdadeiramente competente.

A variedade de títulos e diversidade de habilidades que envolvem o arquiteto de software evidencia a complexidade presente nos estudos acerca das atividades, conhecimentos e deveres esperados para um arquiteto de software. No trabalho descrito em (KRUCHTEN, 2008), Krutchen define os padrões e antipadrões existentes no trabalho do arquiteto de software e propõe uma forma de alocação de tempo com a finalidade de evitar a ocorrência desses antipadrões. Este é um dos principais trabalhos que norteiam e motivam a realização dessa pesquisa. Krutchen apresenta os quatro principais antipadrões arquiteturais: criar uma arquitetura perfeita para o sistema errado, criar uma arquitetura perfeita mas que é muito difícil de implementar, arquitetos que permanecem em sua torre de marfim, afastados da equipe, e arquitetos ausentes. As funções e responsabilidades de um arquiteto de software também são elencadas por Krutchen de acordo com a sua experiência gerenciando uma grande equipe de arquitetura de software em meados da década de 1990 (KRUCHTEN, 2008).

Mesmo em trabalhos mais recentes, como (GALSTER; TAMBURRI; KAZMAN, 2017), os autores ainda continuam corroborando com a ideia de que o papel e as responsabilidades do arquiteto de software geralmente não estão claramente definidos e separados de outras funções no processo de desenvolvimento de software. Pesquisadores e profissionais tentam há anos definir o papel do arquiteto de software, mas ainda não existe um consenso claro sobre o que faz um arquiteto de software, por exemplo, em contraste com um desenvolvedor.

Sendo assim, motivado pela possibilidade de ocorrência dos antipadrões, pluralidade de papéis envolvidos e pela incorreta utilização do termo e do cargo arquiteto de software nas organizações, esse trabalho propõe uma pesquisa com a finalidade de reunir os padrões e antipadrões referentes ao trabalho do arquiteto de software na literatura e verificar se no Brasil o desempenho dos arquitetos de software na prática se aproxima ou se distancia do que foi encontrado na literatura.

## 1.2 Objetivos

O objetivo geral deste trabalho é caracterizar e documentar o papel do arquiteto de software no Brasil com o intuito de avaliar se no Brasil as atividades desempenhadas na prática pelos arquitetos estão de acordo com os padrões ou antipadrões definidos na literatura.

De forma a alcançar o objetivo geral, os seguintes objetivos específicos foram definidos:

- Reunir informações a respeito das definições dos deveres, habilidades e conhecimentos referentes ao papel do arquiteto de software, usando como referência as definições da *Software Engineering Institute* (SEI, 2017b) e de outros artigos escritos por autores já reconhecidos e consolidados na área da arquitetura de software, como em (HOORN et al., 2011), (KRUCHTEN, 2008), (MICROSOFT, 2008) e (KLEIN, 2016).;
- Verificar a situação atual do ensino da Arquitetura de Software como disciplina em nível de graduação nas Instituições de Ensino Superior do Brasil;
- Obter dados sobre as funções desempenhadas na prática pelos arquitetos de software a partir das informações obtidas sobre os deveres, habilidades e conhecimentos referentes ao papel do arquiteto de software;
- Avaliar quantitativamente e qualitativamente os dados das respostas obtidas para caracterizar o papel do arquiteto de software no Brasil;
- Propor um plano de ensino para a disciplina de Arquitetura de Software em nível de graduação;
- Coletar avaliações de professores de engenharia e arquitetura de software para o plano de ensino proposto.

## 1.3 Metodologia

Este trabalho caracteriza-se como uma pesquisa de natureza experimental e possui uma abordagem quantitativa e qualitativa. No intuito de alcançar os objetivos propostos na seção anterior, nesse trabalho são aplicados uma revisão da literatura e um *survey*. Também é realizado um estudo piloto antes do envio final do instrumento de coleta aos participantes. Além disso, é proposto um plano de ensino para a disciplina de Arquitetura de Software em nível de graduação a ser avaliado por professores de engenharia e arquitetura de software.

O objetivo de realizar a revisão da literatura foi buscar trabalhos que auxiliem na obtenção de informações sobre os conhecimentos, funções e deveres de um arquiteto de software, além de como é definido, tratado e implementado na prática o papel do arquiteto de software. As buscas são restritas a artigos escritos após 2008, visto que o trabalho que define os padrões e antipadrões

do arquiteto de software, (KRUCHTEN, 2008), ter sido publicado no referido ano. Somente foram selecionados artigos escritos no idioma inglês. As consultas foram realizadas nas bases digitais IEEE Xplore, ACM Digital Library, Scopus e ScienceDirect.

O objetivo de realizar um estudo piloto é avaliar se o questionário é suficientemente forte para uso no estudo principal, e para melhorá-lo ou refatorá-lo, caso necessário. Além disso, os comentários e observações feitos pelos arquitetos de software envolvidos no estudo piloto foram utilizados para aperfeiçoar ainda mais a pesquisa. Segundo (KITCHENHAM; PFLEEGER, 2002), é essencial que o instrumento seja avaliado. A avaliação é frequentemente chamada de pré-teste e tem vários objetivos diferentes, entre eles:

1. Avaliar se as perguntas são compreensíveis.

2. Avaliar a taxa de resposta provável e a eficácia dos procedimentos de acompanhamento.

3. Avaliar a confiabilidade e validade do instrumento.

4. Garantir que as técnicas de análise de dados correspondam às respostas esperadas.

Segundo (PFLEEGER; KITCHENHAM, 2001), um *survey* não é somente um instrumento (questionário ou *checklist*) para coletar informações. É um sistema abrangente de coleta de informações para descrever, comparar ou explicar conhecimentos, atitudes e comportamentos. Assim, o instrumento de *survey* faz parte de um processo de *survey* maior com atividades claramente definidas:

1. Definição de objetivos específicos e mensuráveis;

2. Planejamento e agendamento da pesquisa;

3. Garantir que os recursos apropriados estejam disponíveis;

4. Projetar o *survey*;

5. Preparar o instrumento de coleta de dados;

6. Validar o instrumento;

7. Selecionar os participantes;

8. Administrar e registrar o instrumento;

9. Analisar os dados;

10. Relatar os resultados.

Neste estudo é realizado um *survey* que utiliza como instrumento de coleta de dados um questionário online enviado por meio de mensagem na caixa de entrada do LinkedIn. Os participantes que receberam o questionário são os profissionais que trabalham no Brasil e que tenham declarado o cargo Arquiteto de Software como função atual na plataforma do LinkedIn.

O questionário contém itens sociodemográficos, como estado de moradia do entrevistado

e experiência (número de anos de trabalho). Esses itens permitem analisar o impacto desses fatores adicionais nas atividades arquiteturais em que os entrevistados estão envolvidos.

Além dos itens sociodemográficos, o questionário, exposto em seguida, é composto por afirmações objetivas que utilizam como padrão de respostas a escala Likert com 5 pontos, sendo: 1- Nunca, 2- Raramente, 3- Às vezes, 4- Muitas vezes e 5- Sempre.

## 1.4 Trabalhos relacionados

No trabalho descrito em (WEINREICH; GROHER, 2016), Weinreich e Groher discutem sobre o papel do arquiteto de software utilizando informações provenientes de entrevistas com um grupo de vinte e cinco profissionais, em média com mais de treze anos de experiência na área. Dentre os profissionais entrevistados estão presentes arquitetos de software, líderes de equipe de software e desenvolvedores seniores com responsabilidades consideradas arquiteturais. Os autores também investigam os fatores que influenciam a tomada de decisões, os papéis e responsabilidades para os diferentes tipos de decisões, a integração do papel do arquiteto de software nas estruturas organizacionais e as ferramentas e práticas utilizadas para gerenciar o conhecimento arquitetural.

Ainda de acordo com Weinreich e Groher (2016), os participantes revelaram que muitos fatores influenciam as diversas maneiras que os arquitetos trabalham e como, quando e por quem as decisões arquiteturais são tomadas. As principais influências potenciais mencionadas foram, respectivamente, os fatores organizacionais, o tamanho da companhia, os fatores de projeto, negócio, individuais, culturais e técnicos. De acordo com as informações coletadas, os autores notaram que a tomada de decisões ainda é a essência da atividade arquitetural, porém, a tarefa principal de um arquiteto de software deixa de ser a tomada solitária de decisões de design, ao passo que as decisões passam a ser um esforço coletivo em que o arquiteto atue como consultor ou gerenciador de conhecimento.

Em (ERDER; PUREUR, 2016), Erder e Pureur dissertam sobre qual o papel evolutivo do arquiteto de software em um novo mundo ágil e centrado em computação na nuvem, que demanda acessibilidade e disponibilidade 24 horas por dia. Para ter sucesso neste mundo e acompanhar o avanço das tecnologias e expectativas, os arquitetos de software devem se afastar de seu papel tradicional e se aproximar de práticas ágeis e contínuas. Os autores elencam alguns aspectos que auxiliam na definição do papel do arquiteto de software, são eles: unidade-chave de trabalho, foco no produto e arquitetura realizada.

A unidade-chave de trabalho do arquiteto de software é definida por Erder e Pureur (2016) como a decisão arquitetural e o conjunto de decisões que são feitas ao longo do desenvolvimento do software. A decisão arquitetural é considerada o resultado mais importante da atividade arquitetural. O papel do arquiteto de software então é conduzir essas decisões para a boa conclusão do projeto, atuando como facilitador para a tomada de decisão e assegurando que

as decisões sejam oportunas. Sobre o foco no produto, os arquitetos devem ter uma visão a longo prazo, pois os produtos e o software sobrevivem mesmo após o fim de um projeto. Os arquitetos devem garantir uma arquitetura coerente e sustentável, para isso as habilidades de comunicação e colaboração tornam-se muito relevantes. A arquitetura realizada é definida pelos autores como a representação do código em execução na infra-estrutura física. O papel do arquiteto torna-se entender, influenciar, melhorar e comunicar essa arquitetura realizada. Por fim, Erder e Pureur (2016) afirmam que além dos arquitetos de software precisarem das habilidades técnicas adequadas, também devem possuir habilidades nas áreas críticas de tomada de decisão, comunicação e colaboração, participação na equipe de entrega e teste e implantação de sistemas.

Em (KLEIN, 2016), John Klein discute sobre o que é necessário para que um arquiteto de software seja considerado um profissional de sucesso. Ao dissertar sobre a definição de arquiteto de software, Klein justifica que as muitas definições existentes de arquitetura de software culminam em diferentes noções do que significa ser um arquiteto de software. O modelo proposto pelo autor possui três funções distintas com diferentes habilidades para os arquitetos de software: *designer* inicial, *extender* e *sustainer*.

A função de *designer* inicial, como o nome sugere, é criar o design inicial do software, identificando os requisitos de qualidade e funcionais arquiteturalmente significativos. A função de papel de *extender* começa depois que o software é entregue ou implantado. Não raramente, os desenvolvedores conseguem elevar rapidamente o valor que o software oferece, expandindo-o ou integrando-o com outros sistemas. Os arquitetos precisam de uma extensa compreensão do software construído para identificar oportunidades e abordagens de integração. A terceira função, *sustainer*, inicia depois que o software está em produção há algum tempo. Neste ponto, o software ainda oferece valor, mas torna-se mais caro para manter e evoluir. A prioridade é evitar a mudança, de modo que o trabalho dos arquitetos é demonstrar a continuidade da relevância e adequação do software à medida que o ambiente muda.

Em (SPINELLIS, 2016), Diomidis Spinellis disserta sobre o papel mutável que o arquiteto de software precisa assumir. Segundo o autor, com as mudanças na indústria de software, o papel do arquiteto também precisa mudar para estar alinhado com o que há de novo. Por exemplo, com a popularização de componentes *o*pen source ou prontos para comprar, torna-se papel do arquiteto equilibrar fatores como risco, qualidade, restrições e custos ao decidir se é mais vantajoso utilizar, comprar ou construir algum componente. De acordo com Spinellis, o arquiteto também deve agir como instrumento para oferecer e gerenciar a agilidade do projeto de software e auxiliar os desenvolvedores com a testabilidade do software, definindo em que momento usar teste unitário, componentes que podem ser testados de forma isolada e testes automatizados. O autor também disserta sobre o impacto das decisões arquiteturais em atributos de qualidade como usabilidade, segurança, confiabilidade e eficiência.

Como pode ser observado, a definição do papel do arquiteto é um tema muito abordado pelos pesquisadores. Essa definição clara torna-se necessária para que o arquiteto de software

saiba quais conhecimentos e habilidades deve possuir e quais funções deve desempenhar. Por falta de definição, não raramente, os arquitetos de software desempenham de forma errônea as suas funções ou atuam em funções que não deveriam ser realizadas por um arquiteto de software. Existem casos em que o título de arquiteto de software é utilizado indevidamente, como por exemplo, apenas para justificar o aumento de salário de um programador experiente (KRUCHTEN, 2008).

## 1.5 Estrutura da dissertação

O texto desta dissertação está organizado em cinco capítulos sucintamente descritos a seguir.

No Capítulo 2 são apresentados os fundamentos teóricos que embasam a realização deste trabalho. Suas subseções trazem conceitos referentes a Arquitetura de Software, Decisões Arquiteturais e o papel do Arquiteto de Software.

No Capítulo 3 é apresentado um estudo sobre o trabalho do arquiteto de software no Brasil. Na Seção 3.1 são discutidos os padrões e antipadrões relacionados ao trabalho do arquiteto de software. Na seção 3.2 é apresentado o desenvolvimento do projeto de *Survey* e como ocorreu sua validação por meio de um estudo piloto. Na Seção 3.3 são discutidas as formas de seleção de participantes e envio de questionários para os mesmos. A análise dos dados é explicada na Seção 3.4. Por fim, na Seção 3.5 são discutidas as ameaças à validade.

No Capítulo 4 é apresentado o processo realizado para o desenvolvimento de uma proposta de disciplina de arquitetura de software em nível de graduação. Inicialmente, na Seção 4.1 é estudado o panorama atual do ensino da arquitetura de software no Brasil. Na seção 4.2 é apresentada a proposta inicial de plano de ensino para a disciplina e a avaliação realizada por professores de engenharia/arquitetura de software que lecionam em IES no Brasil. Por fim, na Seção 4.3 é exposta a versão final do plano de ensino após alterações sugeridas na avaliação dos professores.

No Capítulo 5 são apresentadas as conclusões, principais contribuições, limitações e trabalhos futuros.

# 2 Referencial Teórico

Este Capítulo apresenta as principais teorias e conceitos relacionados a este trabalho. Estes conceitos são necessários para o entendimento e correta interpretação dos resultados desta dissertação. As próximas seções estão organizadas da seguinte maneira: na Seção 2.1 são abordados os principais conceitos referentes a Arquitetura de Software e sua importância para o desenvolvimento de software, a Seção 2.2 trata sobre as Decisões Arquiteturais e, por fim, na Seção 2.3 o Papel do Arquiteto de Software é discutido.

## 2.1 Arquitetura de Software

Segundo Kruchten, Obbink e Stafford (2006), a primeira referência ao termo "arquitetura de software" ocorreu em 1968 em uma conferência sobre engenharia de software promovida pela Organização do Tratado do Atlântico Norte (OTAN). A palavra "arquitetura" era usada principalmente no sentido de arquitetura do sistema (ou seja, a estrutura física de um sistema de computador) ou, às vezes, no sentido mais restrito de um determinado conjunto de instruções da família de computadores. Essa definição permaneceu até o final da década de 1980. O conceito de arquitetura de software como uma disciplina distinta da Engenharia de Software só começou a surgir a partir da década de 1990.

Nas primeiras décadas da Engenharia de Software, a Arquitetura de Software era, em grande parte, um assunto *ad hoc*, ou seja, era tratada como algo provisório que atendia apenas ao propósito de documentar, mesmo de forma incompleta, a estrutura do sistema. Desde o início do desenvolvimento de sistemas de software complexos os projetistas responsáveis por descrever sua estrutura utilizavam diagramas informais de seta e retas, que raramente eram mantidos e atualizados quando o sistema era construído. As escolhas arquiteturais eram feitas de maneira idiossincrática - tipicamente adaptando algum projeto anterior, apropriado ou não para aquele contexto (GARLAN, 2000).

Essas representações estruturais do software eram chamadas de arquiteturas, mas o conhecimento de estilos comuns - isto é, formas estruturais geralmente úteis - não era sistematicamente organizado ou ensinado. Em geral, era impossível analisar a descrição de uma arquitetura de software por coerência ou inferir propriedades não triviais sobre ela. Não havia praticamente nenhuma maneira de verificar se uma determinada implementação de um software representava fielmente seu design arquitetural. No entanto, apesar de sua informalidade, as descrições arquiteturais já eram centrais para o design do software (GARLAN, 2000; SHAW; CLEMENTS, 2006).

Com o passar dos anos a arquitetura de software mudou e evoluiu muito, tornando-se uma

parte reconhecidamente indispensável ao desenvolvimento de um software. Mais especificamente nas últimas duas décadas, a importância e os papéis da arquitetura de software dentro do ciclo de vida do desenvolvimento de software sofreu alterações (VLIET, 2008; SHERMAN; HADAR, 2015). Paradoxalmente, apesar do avanço e amadurecimento da disciplina, ainda não há um consenso sobre uma resposta satisfatória, curta e precisa para uma simples questão - não existe uma definição amplamente aceita do que vem a ser arquitetura de software (KRUCHTEN; OBBINK; STAFFORD, 2006; SHERMAN; HADAR, 2015).

Uma das primeiras formas de descrever a arquitetura de software foi a famosa equação "*{elements, forms, rationale} = software architecture*" introduzida em 1992 por Dewayne Perry e Alexander Wolf no artigo ''Foundations for the Study of Software Architecture" (KRUCHTEN; OBBINK; STAFFORD, 2006). Atualmente, uma das definições de arquitetura de software mais aceitas pode ser encontrada na ISO/IEC/IEEE 42010, norma que trata da descrição da arquitetura no âmbito da Engenharia de Software e Sistemas e especifica a maneira pela qual as descrições de arquitetura dos sistemas são organizadas e expressas. De acordo com o padrão (ISO/IEC/IEEE, 2011), a arquitetura de software é a organização fundamental de um sistema, incorporada em seus componentes, suas relações entre si e com o meio ambiente, e os princípios que regem o seu design e evolução.

Como forma de exemplificar a pluralidade de definições existentes, apenas no site da SEI (*Software Engineering Institute*) (SEI, 2017a) mais de cem definições diferentes podem ser encontradas. O número de definições é tão alto que elas foram divididas em categorias: clássicas, modernas e definições da comunidade.

Em termos gerais, a arquitetura de software pode ser considerada um modelo abstrato de um software que tem o objetivo de satisfazer requisitos de software (TANG; LAU, 2014). A arquitetura de software é responsável por capturar a estrutura do software em termos de componentes e a forma como eles interagem. Ela é um item crítico no projeto e na construção de qualquer sistema de software complexo, pois também é a arquitetura de software que define as regras de design do software e considera como esse software pode mudar e evoluir (GORTON, 2011).

A arquitetura de software desempenha um papel fundamental como uma ponte entre os requisitos funcionais e não-funcionais e a implementação do software. Ao fornecer uma descrição abstrata de um software, a arquitetura de software expõe certas propriedades, ocultando outras. O ideal é que essa representação forneça um guia para o software como um todo, permitindo que os *designers* raciocinem sobre a capacidade de um software satisfazer certos requisitos-chave, captar explicitamente a intenção e os princípios que governam seu design e prescrever um modelo para a construção e composição do software (GARLAN, 2014).

Entre outras aplicações possíveis, a arquitetura de software também é usada para comunicação e documentação, design e raciocínio sobre propriedades importantes do software e como um modelo para a implementação do software. Weinreich e Buchgeher (2012) definem a arquite-

tura de software como uma abstração de um sistema de software, que é perfeitamente adequada para comunicação, raciocínio e aprendizado sobre elementos e propriedades importantes de um software.

Uma boa arquitetura de software pode ajudar a garantir que um software satisfaça requisitos funcionais e de qualidade, incluindo desempenho, confiabilidade, portabilidade, escalabilidade e interoperabilidade. Por sua vez, uma arquitetura de software ruim, ou mal projetada, pode ser desastrosa e causar problemas que podem ser muito caros de resolver no futuro (GARLAN, 2014).

Segundo Hohpe et al. (2016), a arquitetura de software abrange cinco aspectos: contexto, elementos, formas, fundamentação e realização. O arquiteto de software deve possuir conhecimentos em todos eles. O contexto diz respeito aos atributos de qualidade, velocidade e valor do negócio, custo e risco, princípios arquiteturais e gerenciamento da dívida técnica. Os elementos e formas estão relacionados com as notações, dados e relações da arquitetura em tempo de execução e *frameworks*. A fundamentação, no que lhe concerne, inclui o gerenciamento de conhecimento arquitetural e a tomada de decisão nos estágios iniciais. A realização, por sua vez, está relacionada com as práticas ágeis, entrega contínua, dimensão temporal, melhor execução de decisões arquiteturais e ciclos de *feedback* contínuos.

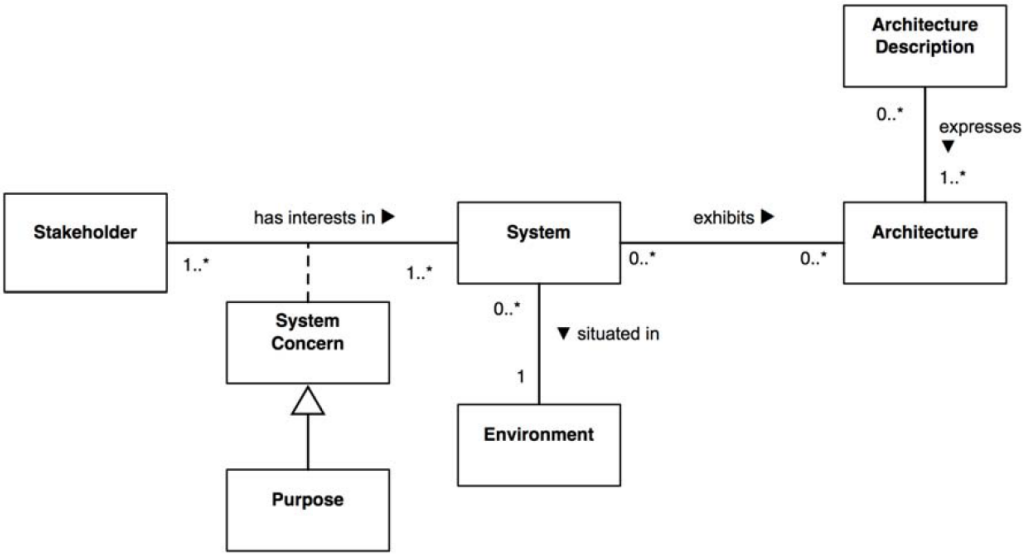
A ISO/IEC/IEEE 42010, já citada anteriormente, apresenta uma forma de representação genérica sobre o contexto da descrição arquitetural que pode ser vista na Figura 2.1. Algumas definições são importantes para o entendimento da mesma:

- Os ***stakeholders*** de um software são as partes interessadas nesse software. Os interesses dos *stakeholders* são expressos como *concerns*;
- Os ***concerns*** são interesses em um software relevante para um ou mais de seus *stakeholders*. Os *stakeholders* possuem propósitos, ou objetivos, a serem atendidos pelo software. Esses propósitos também são classificados como *concerns*;
- O **ambiente** determina a totalidade de influências sobre o software ao longo de seu ciclo de vida, incluindo suas interações com esse ambiente. Um software está situado em um ambiente e esse ambiente pode conter outros softwares.

Os *concerns* geralmente surgem ou se modificam ao longo do ciclo de vida, das necessidades e requisitos do sistema, das escolhas de projeto, e das considerações de implementação e operação. Um *concern* pode se manifestar em muitas formas em relação a uma ou mais necessidades dos *stakeholders*, como por exemplo metas, expectativas, responsabilidades, requisitos, restrições de projeto, premissas, dependências, atributos de qualidade, decisões arquiteturais, riscos ou outras questões pertinentes ao software.

Apesar de inicialmente todo esse contexto parecer complicado, em resumo, os *stakeholders* possuem um ou mais propósitos expressos por meio de *concerns* que motivam a existência

Figura 2.1 – Contexto da descrição arquitetural.




Fonte: ISO/IEC/IEEE (2011).

de um software. Esse, por sua vez, é situado em um ambiente e exibe uma arquitetura de software que pode ser expressa por uma descrição arquitetural.

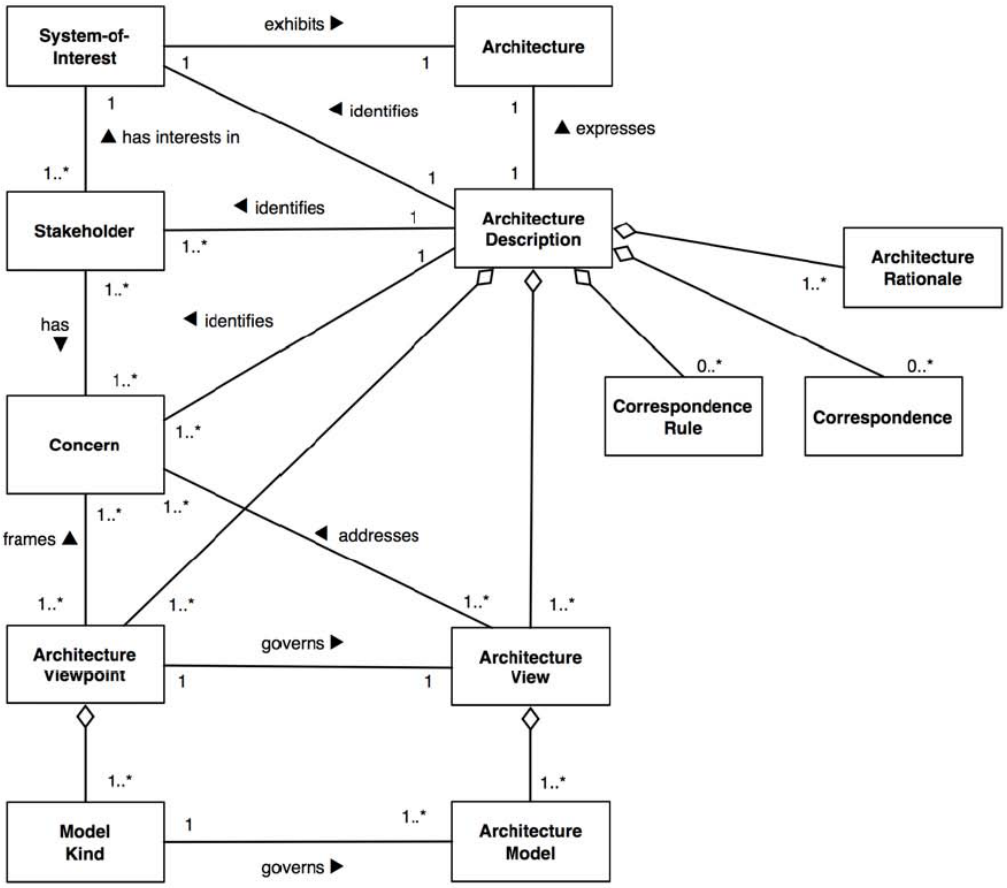
A ISO/IEC/IEEE 42010 também possui uma representação do modelo conceitual de uma descrição arquitetural. Esse modelo pode ser visto na Figura 2.2. Nota-se que uma descrição arquitetural pode incluir uma ou mais visões de arquitetura. Uma visão arquitetural representa o software a partir da perspectiva de um ou mais *concerns* mantidos pelos *stakeholders* do software de interesse. Uma visão arquitetural expressa a arquitetura do software de interesse de acordo com um ponto de vista de arquitetura. Um *concern* pode ser enquadrado por mais de um ponto de vista.

Uma visão é conduzida pelo seu ponto de vista: o ponto de vista estabelece as convenções para construir, interpretar e analisar a visão para tratar dos *concerns* enquadrados por esse ponto de vista. As convenções de ponto de vista podem incluir linguagens, notações, tipos de modelo, regras de design e/ou métodos de modelagem, técnicas de análise e outras operações.

Uma visão da arquitetura é composta por um ou mais modelos arquiteturais. Um modelo arquitetural usa convenções de modelagem apropriadas aos *concerns* a serem tratados. Essas convenções são especificadas pelo tipo de modelo que conduz o modelo arquitetural.

É impossível representar todas as informações relevantes sobre a arquitetura de um software em um único modelo arquitetural, pois cada modelo apresenta apenas uma visão ou perspectiva do software. Uma visão pode refletir como um software é decomposto em módulos, como os processos de *runtime* interagem, ou as diferentes formas como são distribuídos os componentes do software através de uma rede. Tudo isso é útil em momentos diferentes, portanto,

Figura 2.2 – Modelo conceitual de uma descrição arquitetural.




Fonte: ISO/IEC/IEEE (2011).

para ambos, projeto e documentação, geralmente é necessário apresentar múltiplas visões da arquitetura de software (SOMMERVILLE, 2019).

Atualmente a arquitetura de software é reconhecidamente um importante produto ou artefato de software explícito no desenvolvimento de software. Oportunidades de emprego para o cargo de arquiteto de software são amplamente divulgadas em sites de recrutamento, as empresas confiam nas revisões de design arquitetural como pontos críticos e arquitetos de software reconhecem a importância de resolver *tradeoffs* explícitos dentro do espaço de design arquitetural (KRUCHTEN; OBBINK; STAFFORD, 2006; GARLAN, 2014).

A base tecnológica e metodológica do projeto arquitetural evoluiu com o passar dos anos. Quatro avanços importantes foram: (i) a codificação e disseminação de conhecimento em design arquitetural, (ii) o surgimento de plataformas e linhas de produtos e seus ecossistemas associados, (iii) o desenvolvimento de princípios, linguagens e ferramentas para a descrição da arquitetura, (iv) a integração do design arquitetural nos processos mais amplos de desenvolvimento de software e, em particular, a relação entre arquitetura e agilidade (GARLAN, 2014).

## 2.2 Decisões Arquiteturais

Um dos principais processos em arquitetura de software é a tomada de decisões (WEINREICH; GROHER, 2016). Segundo Kruchten (2001), a vida do arquiteto de software pode ser definida como uma longa e rápida sucessão de decisões sub-ótimas parcialmente tomadas no escuro. Isso se deve ao fato de ser exigido do arquiteto de software que decisões sejam tomadas continuamente, inclusive de forma precoce ainda no início do projeto quando pode não haver definições suficientes que fomentem uma decisão acertada.

Em (TOFAN et al., 2014), os autores afirmam que a arquitetura de software é o resultado de um conjunto de decisões arquiteturais. Corroborando com essa afirmação, anteriormente em (JANSEN; BOSCH, 2005), os autores também descrevem a arquitetura de software como um conjunto de decisões de design arquitetural. Os autores definem a decisão de design arquitetural como um conjunto de adições, subtrações e modificações arquiteturais, raciocínio e regras do projeto, restrições e requisitos adicionais que (parcialmente) atendem um ou mais requisitos em uma determinada arquitetura. Os autores ainda afirmam que as decisões de design arquitetural desempenham um papel importante no design, desenvolvimento, integração, evolução e reutilização de arquiteturas de software.

No processo de tomada de decisão arquitetural, os arquitetos de software precisam constantemente equilibrar todos os tipos de restrições, como requisitos e necessidades dos clientes e outros *stakeholders*, limitações tecnológicas e organizacionais e experiência existentes. Nesse equilíbrio ainda é necessário levar em consideração as melhores práticas e padrões arquiteturais. O arquiteto de software, seja como indivíduo, como um papel ou como uma equipe, desenvolve, incrementa e conduz essa arquitetura de software à medida que ela surge, podendo ser composta por milhares de decisões de projeto individuais (BOOCH, 2007).

Segundo Bass, Clements e Kazman (2012) existem sete categorias de decisões de design arquitetural, são elas:

- Atribuição de responsabilidades;
- Modelo de Coordenação;
- Modelo de dados;
- Gestão de recursos;
- Mapeamento entre elementos arquiteturais;
- Decisões sobre o tempo de ligação; e
- Escolha da tecnologia.

Essas categorias não são a única maneira de classificar as decisões de design de arquitetura, mas elas fornecem uma divisão racional de *concerns*. As categorias podem se sobrepor. Porém, não há problema se uma determinada decisão existe em duas categorias diferentes, porque o dever do arquiteto é garantir que todas as decisões são consideradas.

O projeto da arquitetura de software é um processo criativo no qual é projetada uma organização de software capaz de satisfazer aos requisitos funcionais e não funcionais desse software. Por ser um processo criativo, as atividades no âmbito do processo dependem do tipo de software a ser desenvolvido, da formação e experiência do arquiteto de software e os requisitos específicos para aquele software (SOMMERVILLE, 2019).

Uma vez que existem todas essas variáveis que conduzem um software, é útil pensar em projeto de arquitetura de software como uma série de decisões, ao invés de uma sequência de atividades bem definidas. Em geral essas decisões são tomadas pelo arquiteto de software com base no próprio julgamento, que além de tomar as melhores decisões de design arquitetural deve comunicar e motivar essas decisões (HOORN et al., 2011).

Os arquitetos de software sabem que obter uma arquitetura de software correta é um fator crítico de sucesso para o projeto e o desenvolvimento de softwares. Eles reconhecem o valor de fazer escolhas arquiteturais explícitas e aproveitar os designs arquiteturais já utilizados e validados no desenvolvimento de novos produtos (GARLAN, 2014).

Tradicionalmente, os arquitetos de software foram encarregados de tomar decisões que apresentam alto impacto no projeto, seja financeiro, de tempo ou pessoal. São, portanto, decisões cujas alterações não são desejáveis. Como essas decisões frequentemente são feitas no início do projeto, os arquitetos de software utilizam de sua experiência e capacidade de abstração para obtê-las. Porém, o custo final do projeto e os constantes atrasos demonstram que planejar todos os recursos e definir a estrutura logo no início de um projeto é impraticável. A necessidade da resolução desses problemas, juntamente com a demanda crescente por fornecimento de software de alta qualidade mais rapidamente, mudou a forma como as equipes de desenvolvimento se aproximam da tomada de decisões arquiteturais (HOHPE et al., 2016).

Weinreich e Groher (2016) afirmam que o papel do arquiteto de software está deixando de ser o tomador de decisões solitário e passando a ser o gerenciador de conhecimento arquitetural, atuando junto aos *stakeholders* do projeto como facilitador para a tomada de decisões em conjunto e assegurando que as decisões sejam adequadas e coerentes ao contexto onde está inserido.

Uma arquitetura de software complexa provavelmente reflete milhares de decisões, grandes ou pequenas, porém nem todas essas decisões precisam ser documentadas. Tyree e Akerman (2005) afirmam que um arquiteto de software deve tomar decisões que identifiquem os principais elementos estruturais do software, suas propriedades visíveis e suas relações. Para testar a significância arquitetural de uma decisão, um arquiteto de software deve fazer

a seguinte pergunta: esta decisão afeta uma ou mais qualidades do software? Se a resposta for positiva, o arquiteto deve tomar essa decisão e documentá-la. As decisões documentadas facilitam a compreensão geral de um software, compartilhamento de conhecimento arquitetural e os processos de avaliação (KRUCHTEN; CAPILLA; DUEÑAS, 2009).

Os grandes sistemas de software exigem muitas decisões de design arquitetural e, normalmente, muitos *stakeholders* são responsáveis por elas. Algumas decisões são tomadas nos estágios iniciais de um projeto, algumas são feitas durante o projeto e outras podem acontecer até mesmo na implementação. À medida que as decisões são tomadas ao longo do tempo por muitos *stakeholders* diferentes, os tomadores de decisão podem não estar cientes da existência de problemas. Esses problemas podem incluir informações incompletas e definição ambígua. Como resultado desses problemas a qualidade de uma arquitetura de software normalmente acaba sendo afetada (TANG; LAU, 2014).

As decisões de design feitas durante o início do ciclo de vida de um projeto por muitas vezes são difíceis, se não impossíveis, de validar e testar até que partes do software sejam realmente construídas. A prototipagem de componentes-chave arquiteturais pode ajudar a aumentar a confiança em uma abordagem de design, mas às vezes ainda é difícil ter certeza do sucesso de uma escolha em um determinado contexto de aplicação até que o produto de software esteja em execução (GORTON, 2011).

A incerteza aumenta a dificuldade das decisões arquiteturais. Por exemplo, um arquiteto de software pode projetar um novo sistema de software e ao decidir sobre uma tecnologia, pode ser confrontado com a incerteza quanto ao futuro dessa tecnologia, ou, em que medida a tecnologia satisfaz os requisitos. Portanto, abordar a incerteza é importante para decisões arquiteturais (TOFAN et al., 2014).

Em (VLIET; TANG, 2016), os autores explicam que a tomada de decisões de arquitetura de software não é inteiramente racional, e que certas intuições e decisões naturais podem ser influenciadas por emoções e outros fatores cognitivos. Sendo assim, é necessário considerar os fatores humanos que permeiam a tomada da decisão. Um arquiteto de software que projeta um novo software precisa de plena atenção e conhecimentos relevantes para tomar as decisões de design mais corretas (TANG; LAU, 2014; TANG et al., 2017).

Em suas atividades, os arquitetos de software precisam considerar os requisitos funcionais e os atributos de qualidade (ou requisitos não funcionais) dos sistemas de software. O projeto arquitetural de qualquer sistema complexo é conduzido pelos atributos de qualidade, como confiabilidade, disponibilidade, segurança e desempenho. Os arquitetos são frequentemente responsáveis por abordar tais preocupações por meio da tomada de decisões de projeto, que muitas vezes envolve implementar e implantar táticas de arquitetura padrão (por exemplo, auditoria de segurança, gerenciamento de carga para desempenho, redundância ativa para disponibilidade) (TOFAN et al., 2014; REHMAN et al., 2018).

Hans Vliet (2008) reforça que geralmente as decisões não funcionam de forma isolada. Uma decisão de projeto aborda uma ou mais questões relevantes para o problema a ser resolvido. Muito frequentemente, há mais de uma maneira de resolver a questão, de modo que a melhor decisão a ser tomada é a escolha que possui mais características favoráveis dentre as várias alternativas possíveis. Essa escolha particular pode ser responsável por implicações para a tomada de decisões futuras.

As primeiras decisões de projeto restringem as decisões que se seguem e a alteração dessas decisões iniciais tem enormes ramificações. Alterar essas decisões iniciais causará um efeito cascata, em termos das decisões adicionais que deverão ser alteradas. Em determinados momentos a arquitetura deve ser refatorada ou reprojetada, mas essa não é uma tarefa trivial (BASS; CLEMENTS; KAZMAN, 2012). Refatorar ou reprojetar uma arquitetura de um software que já está sendo utilizada em produção é uma tarefa que exige além de um trabalho minucioso, devido a grande quantidade de alterações necessárias, testes exaustivos para garantir que nada implementado na nova arquitetura tenha impacto negativo sobre o que estava funcionando antes da refatoração.

Mesmo sabendo que cada sistema de software é único, sistemas no mesmo domínio de aplicação frequentemente têm arquiteturas similares, que refletem os conceitos fundamentais desse domínio. Por exemplo, linhas de produto de aplicação são os sistemas construídos em torno de uma arquitetura central com variantes que satisfazem aos requisitos específicos do cliente. Ao projetar uma arquitetura de software, o arquiteto precisa decidir o que seu software e as classes de uma aplicação mais gerais têm em comum, e quanto conhecimento dessas arquiteturas de aplicação pode ser reutilizada (SOMMERVILLE, 2019).

Ao passo que novas tecnologias surgem, a arquitetura de software evolui, bem como o papel do arquiteto de software e a forma como ele se aproxima da tomada de decisão arquitetural (GALSTER; TAMBURRI; KAZMAN, 2017). Na próxima seção o papel do arquiteto de software será apresentado.

## 2.3 Papel do Arquiteto de Software

Assim como existem múltiplas definições de o que é uma arquitetura de software, quando se trata do profissional que trabalha diretamente com a arquitetura, o arquiteto de software, também existem várias definições diferentes que tentam sintetizar o seu papel e suas responsabilidades.

De acordo com Galster, Tamburri e Kazman (2017), os arquitetos de software operam em um ambiente social e organizacional com diferentes *stakeholders* e forças que influenciam as tarefas que executam. Além disso, o papel e as responsabilidades muitas vezes não estão claramente separados de outras funções no processo de desenvolvimento de software. Pesquisadores e profissionais tentaram definir o papel da arquitetura de software, mas ainda não existe um

consenso claro sobre o que faz um arquiteto de software, por exemplo, em contraste com um desenvolvedor.

Segundo Klein (2016) o título "arquiteto de software" pode ser dado ao desenvolvedor mais experiente ou ao consultor técnico mais confiável para gerentes. Ou pode, na verdade, referir-se ao membro da equipe que cria e desenvolve a arquitetura do software. Na prática, existem muitas definições de arquitetura de software, cada uma levando a diferentes definições de "arquiteto de software".

Essa diversidade foi refletida em uma pesquisa feita por Clements et al. (2007), motivados pelas interações com arquitetos de software em exercício, onde foi possível observar a complexidade da profissão. Os autores identificaram aproximadamente 200 funções, 100 habilidades e 100 áreas de conhecimento para a prática de arquitetos de software. De forma geral, o raciocínio dos arquitetos de software se concentra em avaliar a probabilidade de que o software implementado atinja os requisitos funcionais e de qualidade para atender aos objetivos do software (KLEIN, 2016).

Para satisfazer o desempenho, a segurança, a confiabilidade e outras questões de qualidade, os arquitetos de software precisam comparar e escolher cuidadosamente uma combinação de padrões arquiteturais, estilos ou táticas. Posteriormente, no ciclo de desenvolvimento, essas opções de arquitetura de software devem ser implementadas de forma completa e correta, para que não haja desvios do design previsto (REHMAN et al., 2018).

Eeles e Cripps (2010) entenderam que os arquitetos de software não apenas são líderes técnicos em projetos de software, mas também são responsáveis pelo sucesso ou fracasso do projeto como um todo. Os autores argumentaram que, para atingir esses objetivos, os arquitetos de software lideram equipes de design de software, monitoram a qualidade do código, testam a cobertura e garantem que o software funcione conforme esperado, compreendem o processo de desenvolvimento e garantem que as equipes envolvidas no desenvolvimento o sigam, compreendem o domínio do negócio, seus conceitos e terminologias, são bons comunicadores, e são tomadores de decisões.

Uma das principais responsabilidades do arquiteto de software é identificar os principais aspectos que dirigirão o design da arquitetura, logo, devem se concentrar na identificação de *concerns* que são arriscados e caros de mudar. Essa experiência técnica é importante porque os arquitetos de software devem tomar decisões técnicas críticas e de longo prazo. Além disso, arquitetos de sucesso devem ter habilidades de liderança e sociais (ANTONINO; MORGENSTERN; KUHN, 2016).

Fowler (2003) propôs duas classificações distintas sobre arquitetos de software:

• *Architectus Reloadus* - é o arquiteto que toma decisões importantes desde o início, garantindo a integridade conceitual de um software e oferecendo um guia para que a equipe possa seguir.

• *Architectus Oryzus* - é o arquiteto que aborda problemas em um projeto colaborando estreitamente com os desenvolvedores, dando suporte e orientando a equipe de desenvolvimento para que ela tenha autonomia de assumir questões mais complexas, agindo dessa forma não corre o risco de se tornar um gargalo arquitetural.

Esses papéis se complementam, ao invés de descrever diferentes abordagens para o mesmo trabalho, e as pessoas são selecionadas para os papéis com base em habilidades e interesses pessoais. O tamanho da empresa também é importante: numa empresa de tamanho menor, o arquiteto pode cobrir um espectro mais amplo de responsabilidades, como, por exemplo, atuar como desenvolvedor líder e assumir um esforço substancial de implementação.

À medida que o sistema de software evolui, os diferentes padrões de investimento e os resultados esperados levam o arquiteto de software a aplicar diferentes conhecimentos e habilidades. Essas habilidades diferem em cada fase do sistema, e uma contribuição bem-sucedida durante uma fase não implica que o arquiteto tenha as habilidades necessárias para ter sucesso na próxima fase da evolução do sistema. Os arquitetos de software geralmente adquirem essas habilidades por meio de uma combinação de experiência prática e treinamento formal (KLEIN, 2005; KLEIN, 2016).

Christensen, Hansen e Schougaard (2009) investigaram os *concerns* do arquiteto de software por meio de uma pesquisa etnográfica em quatro empresas e chegaram às seguintes conclusões:

- Arquitetos de software são portadores de informações; uma grande quantidade de tempo é gasta na interação com os *stakeholders*. Geralmente, os calendários dos arquitetos de software estão cheios de reuniões e apresentações. Até mesmo na rotina diária eles se envolvem em muitas conversas informais com os membros da equipe. Esse trabalho de interação é vital para garantir o alinhamento entre arquitetura e código, metas de negócio e requisitos do cliente;

- A qualidade das pessoas é tão importante quanto a qualidade da estrutura, o que significa que boa comunicação e habilidade de colaboração foram consideradas altamente importantes para o papel do arquiteto de software;

- Muitas atividades dos arquitetos de software são referentes a detalhes arquiteturais relacionados à implementação do código em si. Em três das quatro empresas, o arquiteto codificou regularmente. A quarta empresa insistiu que os arquitetos não deveriam codificar, porém no decorrer do projeto eles o fizeram. Esse fato foi explicado como um erro onde o profissional desviou do papel de arquiteto esperado pela empresa. Em todas as empresas os arquitetos também analisavam o código e discutiam os detalhes da codificação com os desenvolvedores;

- Os arquitetos de software são claramente preocupados com a arquitetura como uma abstração com o intuito de tornar os grandes sistemas de software gerenciáveis. Os arquitetos falaram sobre atributos de qualidade como direcionadores do projeto arquitetural;
- Os arquitetos diferiram entre si em termos de abordagem, segmentados em arquitetos que trabalharam diretamente com projetos de desenvolvimento e arquitetos que se engajaram apenas na arquitetura geral. Os arquitetos também foram classificados em tipos diferentes de arquitetos: “práticos”, orientados a código; e arquitetos “teóricos”, que atuam em maior nível de abstração.

Segundo Galster, Tamburri e Kazman (2017), o ponto real de ter arquitetos é fornecer liderança de pensamento, orientação e transmitir conceitos complexos em termos acessíveis. Isso exige que os arquitetos possuam uma amplitude crescente de conhecimento e capacidades (o arquiteto precisa ser bom desenvolvedor, testador, precisa conhecer tecnologias, precisa entender a operação e a implantação do sistema, precisa se comunicar com o gerenciamento, precisa assumir responsabilidades para a equipe).

No próximo capítulo é apresentado um estudo sobre o trabalho do arquiteto de software no Brasil, buscando compreender quais são as diversas habilidades, papéis e funções desempenhados pelos profissionais no Brasil.

# 3 O Trabalho do Arquiteto de Software no Brasil

Neste Capítulo têm-se a descrição dos conceitos e métodos utilizados para a elaboração e condução do *survey* sobre o trabalho do arquiteto de software no Brasil. Na Seção 3.1, são apresentados padrões e antipadrões do trabalho do arquiteto de software. Na sequência, a Seção 3.2 apresenta a construção do questionário utilizado como instrumento de pesquisa do *survey*, seu objetivo, elaboração, estruturação e condução do estudo piloto. Em seguida, na Seção 3.3, têm-se a etapa da seleção de participantes e envio do questionário. Logo após, na Seção 3.4, têm-se a análise dos dados coletados. Por último, na Seção 3.5, têm-se a descrição das ameaças à validade.

## 3.1 Padrões e Antipadrões no Trabalho do Arquiteto de Software

O conceito de antipadrão foi introduzido por Koenig (1998), definindo um antipadrão como um padrão, exceto que, ao invés de fornecer uma solução, ele fornece algo que parece superficialmente uma solução, mas não é. Nos trabalhos desenvolvidos por Kruchten (2008), Hoorn et al. (2011) e Klein (2016) são definidos antipadrões em relação ao papel do arquiteto de software.

Kruchten (2008) define como os principais antipadrões dos arquitetos de software: (i) criar uma arquitetura perfeita para o sistema errado, (ii) criar uma arquitetura perfeita, mas muito difícil de implementar, (iii) arquitetos em suas torre de marfim, e (iv) os arquitetos ausentes. Esses antipadrões significam, respectivamente: (i) um arquiteto de software que não se comunica e tem uma probabilidade maior de não entender ou apenas compreender parcialmente os interesses dos *stakeholders* para aquele software, (ii) um arquiteto de software que não entende a capacidade da(s) equipe(s) de implementação, antipadrão que criará níveis enormes de estresse e frustração e provavelmente não fornecerá um produto de qualidade a tempo, (iii) arquitetos que estão isolados, não recebendo informações suficientes dos usuários e desenvolvedores e (iv) o arquiteto de software que vem com uma arquitetura completa, fora do contexto, que não se relaciona bem com os problemas reais descritos em muitas organizações.

Hoorn et al. (2011) discorrem sobre o arquiteto de software solitário, caracterizado-o como tomador de decisão solitário que negligencia a documentação e compartilhamento do conhecimento arquitetural. Este comportamento ocasiona a perda de conhecimento arquitetural que serviria para trocar experiências e ideias com colegas, reutilizar as melhores práticas e até

mesmo treinar funcionários juniores. Segundo os autores, os arquitetos preferem permanecer no controle do conhecimento e não se interessam em ferramentas de suporte automatizado ou inteligente para compartilhamento do conhecimento. Porém, quando se trata de consumir conhecimento arquitetural, ferramentas de suporte para a recuperação de conhecimento arquitetural estão entre os itens que despertam interesse. Essa aparente contradição sugere que os arquitetos de software são, em grande parte, tomadores de decisão que preferem passar sua vida profissional em isolamento.

Klein (2016) discorre sobre três antipadrões, ou "padrões de falha" como o autor refere-se na pesquisa, observados em casos em que as habilidades do arquiteto de software não correspondem às necessidades do papel. O primeiro antipadrão é o *"wrap-around"*, um arquiteto responsável apenas pela manutenção de software legado se torna o projetista de um software para substituição. As habilidades de projeto podem estar defasadas pela falta de contato com as práticas de desenvolvimento e a nova arquitetura projetada pode não se encaixar nas habilidades e nos processos da equipe de desenvolvimento, causando atraso e orçamento acima do esperado.

O segundo antipadrão, *"rising star"*, um desenvolvedor que atuou como arquiteto de software durante a fase de integração e obteve sucesso. Com base nesse sucesso, é designado para projetar a arquitetura de um novo sistema de software, sem treinamento formal e sem experiência em projetar no nível arquitetural de um sistema. No entanto, a experiência em desenvolvimento de software ajuda-o a produzir uma arquitetura que se encaixa nas habilidades e processos da equipe de desenvolvimento, porém não satisfaz os principais requisitos funcionais e de qualidade.

Por fim, no antipadrão *"overprotective parent"*, um arquiteto que foi o projetista inicial permanece durante a fase de integração. Para não comprometer a integridade conceitual da arquitetura do software, suas abordagens para expandir o sistema e integrá-lo a outros sistemas propõem mudanças caras e de grande escala. O sistema não pode fornecer valor rapidamente e se torna preterido por haver concorrentes mais ágeis.

Os antipadrões, em sua maioria, sofrerão rejeição, porque os desenvolvedores normalmente devem esperar por decisões de arquitetura de software, então nenhum ou pouco progresso de projeto de arquitetura de software é feito. Dessa forma, torna-se prejudicial para a organização, uma vez que problemas arquiteturais podem causar atrasos, alto custo, software com baixa qualidade, entre outros.

Em relação aos padrões, para este estudo foram coletadas informações sobre definições das funções, habilidades e conhecimento do arquiteto de software, com o intuito de encontrar as atividades consideradas padrões no desempenho da profissão. As principais definições utilizadas como referência foram do *Software Engineering Institute* (SEI, 2017b) e também foi realizada uma revisão de literatura com ênfase principalmente em artigos já reconhecidos sobre os papéis e responsabilidades do arquiteto de software, incluindo (KRUCHTEN, 2008; MICROSOFT, 2008; HOORN et al., 2011; KLEIN, 2016).

Depois de avaliar as informações coletadas, foi reunida uma coleção com 29 atividades que são declaradas pela literatura como padrões que o arquiteto de software deve seguir para ser considerado bem-sucedido. As atividades podem ser vistas na Tabela 3.1, composta pelo número, atividade e o(s) trabalho(s) onde essas atividades foram encontradas.

Essas 29 atividades serviram como base para elaboração do questionário que foi utilizado como instrumento de pesquisa no *survey*.

## 3.2 Projeto do *Survey*

O papel do arquiteto de software tem se tornado cada vez mais comum na indústria de desenvolvimento de software. Títulos de emprego que envolvem o arquiteto de software podem ser encontrados rotineiramente, mas as suas atividades, tarefas e processos diários ainda não são um consenso (GARLAN, 2014; GALSTER; TAMBURRI; KAZMAN, 2017). Por exemplo, arquitetos de software devem desenvolver código-fonte? Como o trabalho do arquiteto de software se relaciona com o trabalho dos gerentes de projeto? Quanto tempo leva para alguém se tornar um verdadeiro arquiteto de software? Essas questões ainda são objeto de debate na academia e na indústria (GARLAN, 2014; BARROSO et al., 2017) e em trabalhos em que padrões e antipadrões foram sugeridos (HOORN et al., 2011; KLEIN, 2016; WOODS; FAIRBANKS, 2018).

Dada a crescente importância da Arquitetura de Software na pesquisa acadêmica e na prática industrial, torna-se necessário compreender qual o papel do Arquiteto de Software. O objetivo desse *survey* é coletar informações sobre quais atividades e comportamentos os arquitetos de software no Brasil realmente necessitam e executam em suas atividades cotidianas na prática, e como isso se aproxima ou se distancia das habilidades, papéis e conhecimentos citados na literatura como essenciais para a sua atuação.

Para elaborar e hospedar o questionário web criado para essa pesquisa, foi escolhido o serviço de formulários online do Google, o *Google Forms*. O questionário é composto por um total de trinta e seis perguntas e é dividido em duas seções. A primeira seção contém seis questões que objetivam reunir dados estatísticos para caracterização dos entrevistados e a segunda parte do questionário concentrou-se em questões sobre as funções desempenhadas pelo arquiteto de software.

A segunda seção visa reunir informações sobre as atividades desempenhadas pelos arquitetos de software, ela contém uma pergunta base e comum para todas as outras questões: “Em que medida as atividades seguintes se encaixam dentro das atividades praticadas no seu emprego como arquiteto(a) de software?”.

Essa segunda parte do questionário é composta por vinte e nove questões em forma de sentenças afirmativas, em que cada sentença representa cada um dos padrões encontrado

Quadro 3.1 – Atividades do Arquiteto do Software

| Nº | Atividade | Autor(es) |
|---|---|---|
| 1 | Definir Arquitetura de Software | Kruchten (2008), SEI (2017b) |
| 2 | Criar documentos que descrevem a arquitetura | SEI (2017b), Hoorn et al. (2011) |
| 3 | Agir como um comunicador da arquitetura de software | SEI (2017b), Hoorn et al. (2011) |
| 4 | Certificar que todos estão usando a arquitetura proposta | SEI (2017b) |
| 5 | Certificar que todos estão usando a arquitetura corretamente | SEI (2017b) |
| 6 | Certificar que a arquitetura saia em etapas de maneira oportuna para que a organização como um toda possa progredir antes que seja concluída | SEI (2017b) |
| 7 | Certificar que o software e a arquitetura estão em sincronia | SEI (2017b) |
| 8 | Certificar que a gerência entende os detalhes arquiteturais necessários | SEI (2017b), Hoorn et al. (2011) |
| 9 | Certificar que a modelagem está sendo feita corretamente e que os atributos de qualidade serão atendidos | SEI (2017b) |
| 10 | Auxiliar na seleção de ferramentas e ambientes | SEI (2017b) |
| 11 | Identificar e interagir com os *stakeholders* para garantir que suas necessidades sejam atendidas | SEI (2017b) |
| 12 | Convencer os *stakeholders* sobre o valor da solução arquitetural | Hoorn et al. (2011), Klein (2016) |
| 13 | Certificar que a arquitetura não é apenas adequada para operações, mas também para implantação e manutenção | SEI (2017b), Klein (2016) |
| 14 | Realizar resolução de conflitos e analisar *tradeoffs*. | SEI (2017b) Microsoft (2008) Klein (2016) |
| 15 | Resolver problemas técnicos | SEI (2017b) |
| 16 | Manter e elevar a moral da equipe de arquitetura | SEI (2017b) |
| 17 | Compreender e planejar a evolução da arquitetura | SEI (2017b), Klein (2016) |
| 18 | Planejar a inserção de novas tecnologias | SEI (2017b) |
| 19 | Gerenciar estratégias para identificar e mitigar riscos associados à arquitetura | SEI (2017b) |
| 20 | Ajudar na compreensão do domínio do problema | Microsoft (2008) |
| 21 | Aplicar padrões já utilizados para resolver problemas anteriormente | Microsoft (2008) |
| 22 | Manter-se atualizado com as últimas tendências | Hoorn et al. (2011) |
| 23 | Participar no planejamento de projetos | Kruchten (2008) |
| 24 | Gerenciar indivíduos e equipes | Microsoft (2008) |
| 25 | Estabelecer decisões arquiteturais | Microsoft (2008) |
| 26 | Pensar no impacto que as decisões têm na arquitetura atual | Hoorn et al. (2011) |
| 27 | Agir como consultor de tecnologia | Microsoft (2008) |
| 28 | Envolver-se em todas as etapas do ciclo de vida do software | Microsoft (2008), Klein (2016) |
| 29 | Auxiliar no *marketing* e definição de futuros produtos | Kruchten (2008) |

na revisão da literatura, catalogados na seção anterior. Por exemplo: “Eu defino a arquitetura de software” que corresponde ao padrão “Definir Arquitetura de Software”. Como opções de resposta foi utilizado o padrão da escala Likert (LIKERT, 1932) com 5 pontos, sendo: 1 - Nunca, 2 - Raramente, 3 - Às vezes, 4 - Muitas vezes e 5 - Sempre.

Todas as trinta e cinco questões, somando as partes I e II do questionário são obrigatórias, porém como complemento foi criada uma trigésima sexta questão opcional e subjetiva com o intuito de reunir comentários sobre a rotina como arquiteto de software e/ou sugestões para a pesquisa.

O primeiro esboço do instrumento de pesquisa foi criado e enviado para quatro arquitetos de software, todos com mais de cinco anos de experiência na área, a fim de avaliar o entendimento das questões e sugerir possíveis melhorias ao questionário. O *Google Forms* foi a plataforma utilizada durante a criação do esboço, plataforma também utilizada para o envio final, dessa forma foi possível manter o padrão de layout mais próximo possível da versão final do questionário. Os arquitetos de software que participaram da pesquisa na fase do estudo piloto não foram incluídos na seleção de participantes que receberam o questionário final, a fim de evitar que o resultado fosse comprometido.

Após o recebimento de todos os *feedbacks*, foi avaliado o que de fato poderia ser modificado e foram realizadas as alterações. Dentre as modificações ocorreu a alteração do texto da afirmação “Eu me certifico que todos estão utilizando a arquitetura e utilizando-a corretamente” que foi transformada em duas afirmações separadas: “Eu me certifico que todos estão utilizando a arquitetura” e “Eu me certifico que todos estão utilizando a arquitetura corretamente”, e a retirada de afirmações que tinham sentido similar a outras ou fugiam do objetivo da pesquisa, são elas: “Eu atuo como emissário(a) da arquitetura”, “Eu possuo acúmulo de conhecimento técnico”, “Eu possuo habilidades em gerência de projetos” e “Eu possuo habilidades de comunicação”.

Depois que as alterações foram realizadas, gerou-se um segundo esboço que foi enviado novamente para os arquitetos de software solicitando uma nova validação. Dessa vez os *feedbacks* foram todos positivos e o esboço se transformou na versão final e pronta para ser enviada aos participantes.

## 3.3 Seleção de Participantes e Envio

O público-alvo escolhido para essa pesquisa foram os profissionais que trabalham ou trabalharam em algum momento de suas carreiras como Arquitetos de Software. Visto que o intuito era alcançar o maior número possível de profissionais espalhados por todo o país, o meio utilizado para o envio dos questionários que pareceu mais promissor em termos de alcance foi a rede de relacionamento profissional chamada LinkedIn.

Para encontrar os potenciais participantes para o estudo foram realizadas três buscas

utilizando as strings “Arquiteto de Software”, “Arquiteta de Software” e “*Software Architect*” dentro da caixa de pesquisa de pessoas da ferramenta, além do filtro localidade estar preenchido com a opção Brasil. As consultas retornaram respectivamente 2092, 36 e 2832 perfis, porém por observação foi possível perceber que muitos perfis estavam presentes em mais de um resultado, logo o número total de profissionais é menor do que a soma dos três resultados. As consultas foram realizadas no dia 23 de Julho de 2018.

Ao iniciar o envio dos questionário via mensagem, foi detectado que o LinkedIn não permite o envio ilimitado de mensagens via caixa de entrada entre pessoas que não estão conectadas, por esse motivo após utilizar a pequena cota de mensagens oferecida pela ferramenta, foi necessário buscar outras redes sociais daqueles indivíduos, logo alguns questionários foram enviados por meio de outras plataformas que estavam relacionadas como forma de contato dentro do perfil, como E-mail, Facebook ou Twitter.

A lentidão provocada por esse processo multiplataforma continuou até o momento em que foi descoberto que o convite individual para a rede de contatos do LinkedIn permite enviar junto um texto de 300 caracteres, portanto, foi necessário resumir o texto do convite original com 1253 caracteres (Apêndice A) contendo identificação, objetivo do estudo e o pedido para participação da pesquisa, dentro dessa limitada quantidade de caracteres (Apêndice B).

Para enviar o convite foi necessário abrir cada perfil individualmente, clicar na opção “*Connect*” e depois em “*Add Note*”, inserir a mensagem solicitando a resposta ao questionário e por fim clicar em “*Send Invitation*”. O envio dos questionários foi realizado entre agosto e setembro de 2018. Inicialmente, foram enviados questionários para 3187 profissionais que declararam entre as suas posições do LinkedIn terem exercido a profissão de arquiteto de software no Brasil.

A participação na pesquisa foi voluntária. Todos os respondentes convidados foram informados sobre a pesquisa com um texto de convite descrevendo as metas do estudo. Além dos 3187 convites via redes sociais, o questionário também foi enviado ao comitê de arquitetura de software, composto por 45 membros, de uma das maiores empresas multinacionais de software do Brasil.

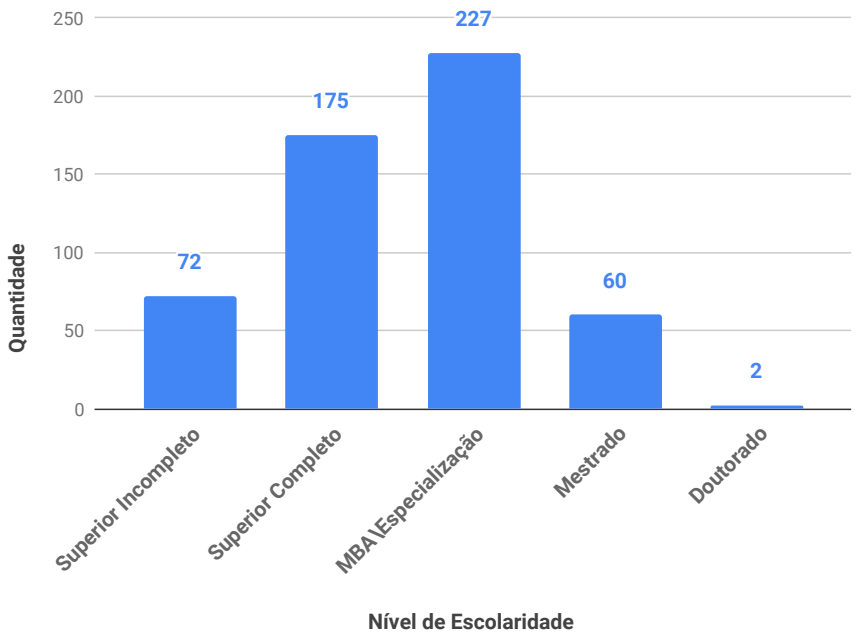
Assim, somando-se o convite enviado ao comitê de arquitetura de software, chegou-se a um total de 3232 convites, dos quais foram obtidas 536 respostas, alcançando uma taxa de resposta de aproximadamente 16,6%. Do total das 536 respostas, 72 participantes também responderam à questão subjetiva descrevendo sobre particularidades da sua vivência como arquitetos de software e dificuldades encontradas no exercício da profissão.

## 3.4 Análise dos Dados

Nesta seção, os resultados do *survey* são apresentados. Primeiro são apresentados os resultados das questões de caracterização. Para cada questão, foi criado um gráfico com a quantidade de respostas referentes a cada alternativa. Em seguida, discorre-se sobre as questões referentes às atividades, funções, habilidades e comportamentos do Arquiteto de Software.

A Figura 3.1 descreve o quantitativo de respostas referentes a questão sobre o nível de educação dos participantes. Nota-se que a maioria dos participantes possui um MBA ou outro curso de pós-graduação (42,4%). Este é um fato importante, pois é essencial aos arquitetos de software que eles possuam uma bagagem de conhecimento e mantenham-se atualizados com as novas tendências e muitos deles aparentemente tem buscado essa melhor qualificação por meio de um MBA profissional.

Figura 3.1 – Nível de Escolaridade.




Uma grande parte dos participantes do estudo (86,6%) completou ao menos um curso de graduação, principalmente em áreas como Ciência da Computação, Sistemas de Informação ou Engenharia de Computação, desses, 32,5% cursaram somente a graduação, 42,4%, como já citado, cursaram também um MBA ou especialização, 11,9% chegaram a completar um mestrado acadêmico em Ciência da Computação e apenas 0,37% cursaram um doutorado. O que chama atenção para esses números é a quantidade de participantes do estudo que ainda frequentam o curso de graduação (13,4%) e o pequeno número de pessoas com doutorado, apenas 2.

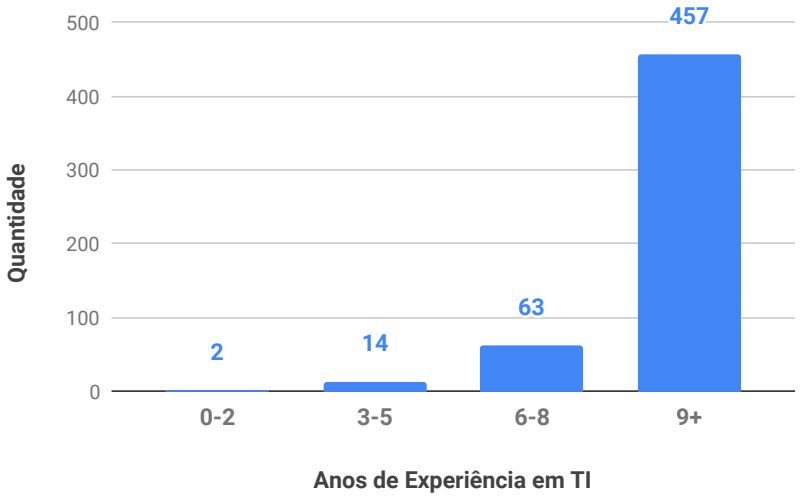
Esse número relevante de profissionais que não tem nível superior mas atuam como arquitetos de software pode refletir o fato de que no Brasil ainda não existe um conselho ou regulamentação oficial sobre o exercício das profissões de Tecnologia da Informação (TI), logo,

pode se tratar de um profissional com experiência na área mas que optou em não cursar um nível superior. Em relação a baixa porcentagem de doutores, isso pode ser justificado pelo fato da arquitetura ser uma área muito técnica e no Brasil a cultura do doutorado ser mais voltada para a área acadêmica, logo pode não ter se tornado uma opção de profissionalização atraente aos Arquitetos de Software. As causas e consequências desses fatos podem ser melhor investigados em trabalhos futuros.

A Figura 3.2 reflete a situação referente a questão sobre o número de anos de experiência em TI antes de se tornar um arquiteto de software. Como pode ser visto, 85,3% dos entrevistados têm mais de 9 anos de experiência e 11,8% têm entre 6 e 8 anos, ou seja, 97% dos entrevistados têm mais de 5 anos de experiência em TI.

Essa concentração de respostas de profissionais com mais de 9 anos de experiência aponta que geralmente os arquitetos de software trabalham na indústria de desenvolvimento de software há muito tempo e já têm experiência pessoal com TI, provavelmente evoluindo por meio de outros cargos até alcançar a posição de arquiteto de software. Dos 3% restantes, 2,6% têm entre 3 e 5 anos de experiência e apenas 0,4% têm menos de 2 anos de experiência, confirmando que são raros os casos em que o profissional já adentra no mercado de trabalho ocupando a posição de arquiteto de software. Assim como afirma um dos participantes em seu depoimento: *"Eu trabalho como arquiteto de software tem pouco tempo, minha carreira teve uma evolução muito rápida em relação aos cargos e outros profissionais da área [...]"*.

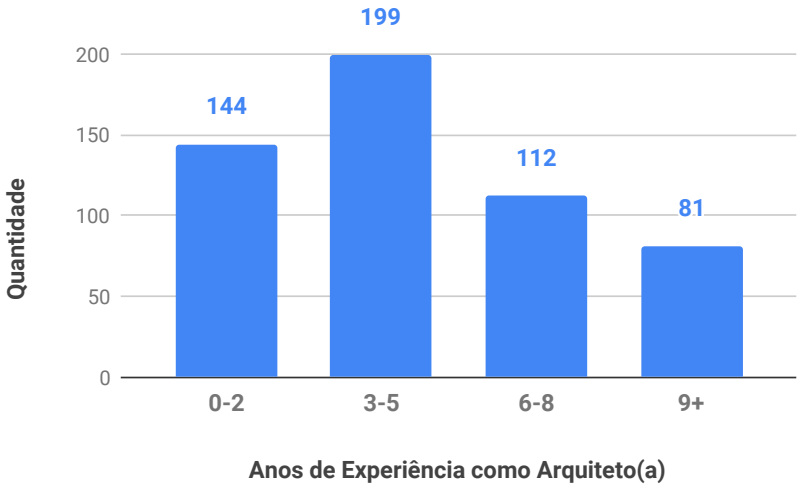
Figura 3.2 – Anos de Experiência em TI.




A Figura 3.3 apresenta o número de anos de experiência na função de arquiteto de software. Nesse quesito todas as alternativas obtiveram uma quantidade significativa de respostas. Mais de um quarto dos participantes (26,9%) têm menos de dois anos de experiência, ou seja, estão no início de suas carreiras como arquitetos de software. Em contraste, 15,1% dos participantes do estudo são arquitetos de software experientes, com mais de 9 anos atuando no cargo. A alternativa com a quantidade de respostas mais expressiva, com 37,1%, foi de arquitetos

de software com 3 a 5 anos de experiência, e os 20,9% restantes são arquitetos de software com 6 a 8 anos de experiência. Essa diversidade é relevante ao estudo que dessa forma consegue alcançar um público em diferentes níveis de maturidade dentro da profissão.

Figura 3.3 – Anos de Experiência como Arquiteto de Software.

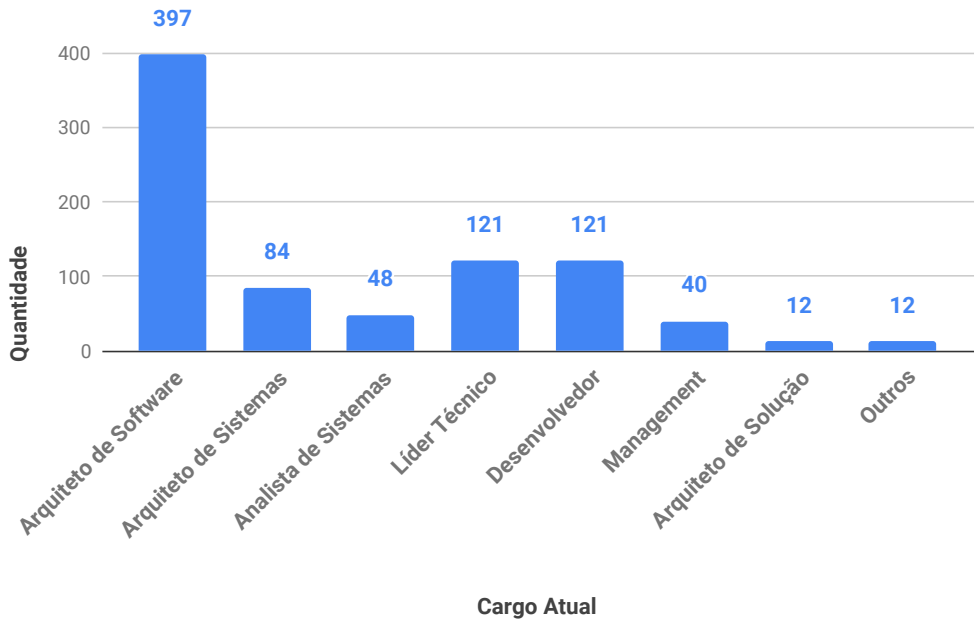



A Figura 3.4 ilustra a posição atual dos participantes. De acordo com a filtragem realizada no LinkedIn, sabe-se que todos os participantes trabalham ou trabalharam em algum momento de suas carreiras como arquiteto de software. Essa questão tem como objetivo descobrir quantos deles ainda estão atuando como arquitetos de software, e caso não atuem mais como arquitetos de software qual a sua posição atual. Também há o intuito de cobrir os casos em que as pessoas atuam como arquitetos de software mas não estão posicionados como tal em seu trabalho atual. Como sugerido em (GORTON, 2011), pode haver várias nomenclaturas para a posição de arquiteto de software. É difícil distinguir quais competências diferenciam um trabalho do outro.

Visto que os profissionais podem possuir mais de um emprego e desempenhar papéis diferentes em cada um deles, foi escolhida a opção de múltipla escolha para essa questão em específico, portanto a soma das porcentagens excede 100%. A maioria dos entrevistados (74,0%) trabalha atualmente como arquitetos de software, 15,7% trabalham como arquitetos de sistemas, 18,5% atuam como desenvolvedores de software, 22,6% trabalham como líder técnico e uma minoria trabalha em outras áreas como Análise de Sistemas e Gerencial.

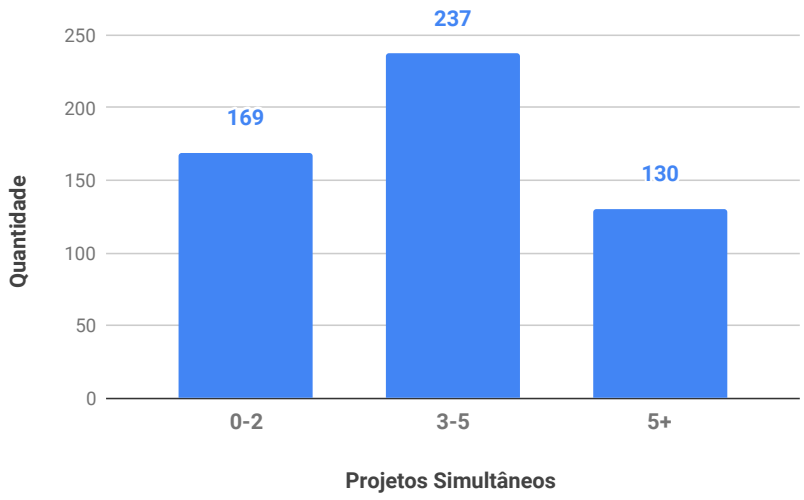
Por meio desses números pode-se notar que muitos arquitetos de software acabam por atuar também como arquitetos de sistemas ou arquitetos de solução, entre outros, tornando difícil definir o que diferencia esses cargos. Assim como é afirmado em uma das respostas subjetivas coletadas na pesquisa: "*Coloquei o cargo arquiteto de soluções, mas basicamente faço o mesmo que quando era arquiteto de software/sistemas. Creio ser apenas nomenclaturas entre empresas*". Muitos dos participantes também acabaram por atuar mais na área técnica, se tornando desenvolvedores ou líderes técnicos, profissões em que certamente o conhecimento adquirido como arquiteto é muito útil.

Figura 3.4 – Cargo Atual dos Participantes.




A Figura 3.5 descreve em quantos projetos os participantes estão envolvidos simultaneamente no seu trabalho atual. Para essa questão, todas as alternativas também obtiveram uma quantidade significativa de respostas. Cerca de um terço (31,5%) dos entrevistados estão envolvidos com no máximo 2 projetos simultaneamente, infere-se que uma parte deles pode estar trabalhando com apenas um projeto. 44,2% estão envolvidos simultaneamente com 3 a 5 projetos e 24,3% estão envolvidos com mais de 5 projetos simultaneamente.

Figura 3.5 – Quantidade de Projetos Simultâneos.

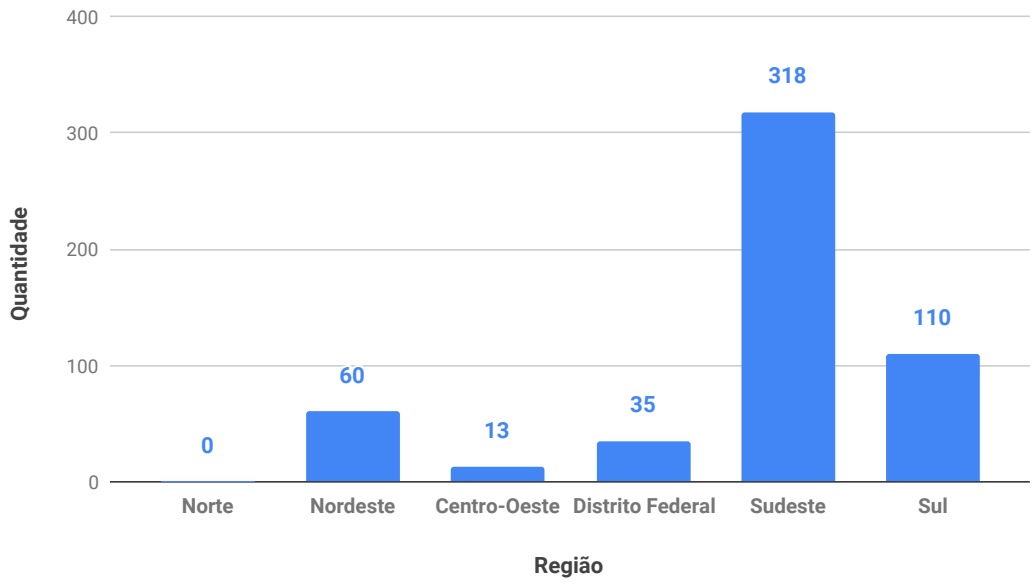



Essa grande variação entre um ou vários projetos pode ser justificada por vários fatores: existência ou não de uma equipe de arquitetura, pelo tamanho da empresa, pela complexidade dos projetos envolvidos, entre outros. Um grande projeto que envolve muitas pessoas pode

exigir mais de um arquiteto de software do que dois pequenos projetos que já possuam soluções similares. É esperado do arquiteto do software que ele possa atuar em várias frentes em diferentes projetos, como exemplo tem-se essa observação feita por um dos participantes: *"Atualmente trabalho com 4 projetos simultâneos, e meu papel como arquiteto de software varia nesses projetos, e essa variação ocorre principalmente devido a maturidade do time, onde necessito atuar mais nas decisões arquiteturais e técnicas, em times mais experientes é possível delegar mais essas decisões."*.

A Figura 3.6 descreve em qual região do Brasil os participantes trabalham atualmente. Dentre as alternativas estão presentes as cinco regiões do Brasil e o Distrito Federal. Para essa questão houve uma diferença significativa entre as respostas. Nenhum dos participantes trabalha na região Norte do país, enquanto que 61,62% dos participantes trabalha na região Sudeste e 21,31% na região Sul. Para as regiões Nordeste, Centro-Oeste e Distrito Federal a quantidade de respostas foi, respectivamente, 11,62%, 2,51% e 6,78%.

Figura 3.6 – Região do Brasil onde trabalham.




A quantidade de respostas concentradas na região Sudeste do Brasil deve-se ao fato que a região Sudeste é considerada o maior mercado tecnológico do país, onde se concentram a maioria das empresas de Tecnologia da Informação e consequentemente, onde atua um maior número de profissionais da área.

Para representar os resultados da seção II do questionário foi seguida uma abordagem diferente, uma vez que todas as questões tem as mesmas alternativas de respostas, as atividades mais desempenhadas pelos arquitetos de software no seu trabalho foram reunidas na Tabela 3.1. Essa tabela é composta do número da questão, a porcentagem de participantes que escolheram, respectivamente, as alternativas "Sempre" e "Muitas Vezes", e a soma das porcentagens dessas duas alternativas, em ordem decrescente. Para este estudo, a soma das alternativas "Sempre" e

"Muitas Vezes" é considerada como o total de respostas positivas da questão. A única questão que não está presente nessa tabela é a questão 29, "Eu auxilio no marketing e definição de futuros produtos" uma vez que a soma das alternativas "Sempre" e "Muitas vezes" é apenas 17,70%.

Como pode ser visto na Tabela 3.1, as cinco atividades mais executadas pelos arquitetos de software no Brasil foram: "Eu penso no impacto que as minhas decisões têm sobre a arquitetura atual" (26), "Eu possuo habilidade para aplicar padrões já usados na resolução de problemas semelhantes" (21), "Eu me certifico de que a arquitetura não é apenas certa para operações, mas também para implantação e manutenção" (13), "Eu compreendo o domínio do problema" (20) e "Eu resolvo problemas técnicos" (15).

Tabela 3.1 – Total de Respostas Positivas.

| Nº Questão | Sempre | Muitas Vezes | Sempre+Muitas Vezes |
|---|---|---|---|
| 26 | 67,20% | 28,50% | 95,70% |
| 21 | 39,20% | 50,40% | 89,60% |
| 13 | 51,10% | 35,60% | 86,70% |
| 20 | 38,60% | 47,60% | 86,20% |
| 15 | 42,70% | 42,00% | 84,70% |
| 22 | 45,70% | 38,80% | 84,50% |
| 25 | 32,50% | 51,70% | 84,20% |
| 10 | 43,70% | 39,70% | 83,40% |
| 17 | 36,90% | 43,70% | 80,60% |
| 1 | 34,90% | 45,70% | 80,60% |
| 3 | 38,40% | 39,20% | 77,60% |
| 4 | 35,60% | 40,90% | 76,50% |
| 7 | 30,60% | 45,50% | 76,10% |
| 18 | 35,80% | 38,50% | 74,30% |
| 27 | 32,10% | 41,80% | 73,90% |
| 11 | 33,00% | 39,20% | 72,20% |
| 5 | 30,00% | 40,70% | 70,70% |
| 9 | 28,70% | 41,80% | 70,50% |
| 23 | 29,90% | 39,70% | 69,60% |
| 16 | 31,00% | 36,60% | 67,60% |
| 19 | 23,50% | 43,70% | 67,20% |
| 14 | 20,30% | 43,80% | 64,10% |
| 12 | 24,80% | 38,80% | 63,60% |
| 28 | 23,10% | 38,20% | 61,30% |
| 8 | 24,10% | 35,30% | 59,40% |
| 6 | 20,10% | 38,60% | 58,70% |
| 2 | 19,40% | 35,40% | 54,80% |
| 24 | 21,50% | 26,70% | 48,20% |

Tendo em mente que é necessário que o arquiteto de software encontre um equilíbrio entre conhecimento técnico, conhecimento de domínio e habilidades de comunicação (KRUCHTEN, 2008; CORREA, 2013), este conjunto de atividades corrobora fortemente com o que se espera

Tabela 3.2 – Total de Respostas Negativas

| Nº Questão | Às vezes | Raramente | Nunca | Total Negativas |
|---|---|---|---|---|
| 29 | 29,7% | 32,5% | 20,1% | 82,3% |
| 24 | 29,4% | 16,4% | 6% | 51,8% |
| 2 | 33,6% | 10,3% | 1,3% | 45,2% |
| 6 | 29,1% | 9,3% | 2,8% | 41,2% |
| 8 | 27,8% | 9,3% | 3,5% | 40,6% |
| 28 | 26,4% | 10,6% | 1,7% | 38,7% |
| 12 | 25,9% | 7,8% | 2,6% | 36,3% |
| 14 | 27,6% | 6,2% | 2,1% | 35,9% |
| 19 | 23,7% | 7,3% | 1,9% | 32,9% |
| 16 | 23,7% | 6,9% | 1,9% | 32,5% |
| 23 | 23,7% | 6% | 0,7% | 30,4% |
| 9 | 20,3% | 7,1% | 2,1% | 29,5% |
| 5 | 22,6% | 6% | 0,7% | 29,3% |
| 11 | 21,8% | 5% | 0,9% | 27,7% |
| 27 | 16,8% | 6,7% | 2,6% | 26,1% |
| 18 | 19,4% | 5,4% | 0,6% | 25,04% |

de um arquiteto de software.

Com intuito semelhante ao da Tabela 3.1, foi criada a Tabela 3.2 que por sua vez reúne as atividades menos realizadas pelos profissionais em seu trabalho. A Tabela 3.2 é composta do número da questão, a porcentagem de participantes que escolheram, respectivamente, as alternativas “Às vezes”, “Raramente” e “Nunca”, e a soma das porcentagens dessas alternativas, em ordem decrescente. Foram selecionadas apenas as questões que têm mais de 25% de taxa de resposta negativa somando as alternativas “Raramente”, “Nunca” e “Algumas vezes”, considerando-as mais relevantes nesse critério.

As cinco atividades menos executadas pelos arquitetos de software no Brasil foram: “Eu auxilio no marketing e futuras definições de produtos” (29), “Eu gerencio indivíduos e equipes” (24), “Eu crio documentos que descrevem a arquitetura de software” (2), “Eu me certifico que a arquitetura seja definida em etapas para que haja progresso na organização antes que a arquitetura seja completamente definida” (6) e “Eu me certifico que a administração entende os detalhes necessários da arquitetura” (8).

Conforme apresentado na Tabela 3.1, a questão 26 apresentou a maior taxa de resposta para a alternativa “Sempre” (67,20%) e ficou em primeiro lugar no ranking das atividades, totalizando 95,70% de respostas positivas. Um dos principais processos na arquitetura de software é a tomada de decisão (KRUCHTEN, 2001; WEINREICH; GROHER, 2016). Portanto, essa pergunta específica com uma alta taxa de respostas positivas significa que os arquitetos de software no Brasil aparentemente estão preocupados com as consequências futuras que suas decisões arquiteturais podem causar durante o desenvolvimento e no uso futuro do software.

Apenas 67,20% dos entrevistados disseram que sempre ou muitas vezes criam estratégias de identificação e mitigação de riscos associados à arquitetura de software. De acordo com (SPINELLIS, 2016), tornou-se papel do arquiteto de software equilibrar fatores como risco, qualidade, restrições e custos ao decidir se é mais vantajoso usar, comprar ou construir um componente, então para que essa questão refletisse essa preocupação seria ideal que ela possuísse maior aderência por meio de respostas positivas.

Na questão 20, "Eu compreendo o domínio do problema", o índice de respostas positivas foi alto, 86,20%. É importante que o arquiteto de software compreenda muito bem o domínio do problema para que ele escolha os detalhes arquiteturais que representem a melhor solução para cada caso.

Dentro do tema decisões arquiteturais, 84,20% dos entrevistados afirmaram na questão 25 que sempre ou muitas vezes "Estabelecem decisões arquiteturais". Esse é um bom resultado, uma vez que a arquitetura de software é o resultado de um conjunto de decisões arquiteturais (TOFAN et al., 2014), e que cada vez mais o arquiteto de software tem se tornado o gestor desse conhecimento, atuando como facilitador para tomada de decisões e garantindo que as decisões são apropriadas e coerentes com o contexto de ação (WEINREICH; GROHER, 2016).

A questão 13 também foi uma das que recebeu as maiores taxas de respostas positivas (86,70%), "Eu me certifico de que a arquitetura não é apenas certa para operações, mas também para implantação e manutenção". Esse item é relevante, uma vez que a arquitetura de software está em constante evolução e adaptação. De acordo com Erder e Pureur (2016), os arquitetos devem ter uma visão de longo prazo, porque os produtos e o software sobrevivem mesmo após o término de um projeto. Não são raros os casos em que mesmo depois que o software é implantado e está sendo utilizado pelo cliente, ainda pode haver requisições de novas funcionalidades, correções ou manutenções.

A respeito da questão 17, "Eu compreendo e planejo a evolução da arquitetura", 80,60% dos respondentes escolheram alternativas sempre ou muitas vezes, um número muito bom, uma vez que é esperado do arquiteto de software que ele planeje a arquitetura já levando em consideração as possíveis evoluções e mudanças no futuro do software.

89,90% dos respondentes também responderam positivamente à questão 21, "Eu aplico padrões já usados na resolução de problemas semelhantes". Isso significa que quase 90% dos arquitetos de software já têm experiência e já participaram de outros projetos anteriormente. Esse é um resultado considerável, porque, como a maioria das decisões arquiteturais tomadas pelos arquitetos de software é feita no início do projeto, eles utilizam sua experiência e habilidades de abstração para obtê-las (HOHPE et al., 2016).

Embora os entrevistados pareçam ser em sua maioria profissionais experientes, eles não pararam de procurar por atualizações, 86,20% disseram que sempre ou muitas vezes estão atualizados das últimas tendências em arquitetura de software (questão 22), e 74,3% dos en-

trevistados sempre ou com frequência planejam a inserção de novas tecnologias (questão 18). Mesmo com uma diferença de quase 12% entre as duas questões, isso significa que mesmo os arquitetos de software mais experientes estão em busca de atualização e tentando implantar as novas tendências que se mostram promissoras.

Dos entrevistados, 76,50% responderam positivamente à questão 4, certificam-se de que todos estão usando a arquitetura de software proposta, porém 18,1% mencionaram que eles se certificam apenas algumas vezes. Consequentemente, com quase a mesma porcentagem de respostas positivas da questão anterior (76,10%) está a questão 7, "Eu me certifico que o software e a arquitetura estão em sincronia". Essa semelhança entre elas faz muito sentido, ao passo que, para que software e sua arquitetura permaneçam sempre sincronizados, é necessário que o arquiteto de software verifique constantemente se a arquitetura de software proposta está realmente sendo usada pelos produtos de software.

Quando se trata de certificar que a arquitetura de software proposta além de ser usada, está sendo usada corretamente (questão 5), o resultado foi diferente. 70,70% dos entrevistados responderam que realizam esta atividade sempre ou muitas vezes. Como o número de respostas positivas foi menor do que a questão anterior, "Eu me certifico que o software e a arquitetura estão em sincronia", o número de respostas negativas também aumentou. 22,60% dos respondentes disseram certificar-se disso apenas algumas vezes, e 6,0% responderam que raramente certificam que a arquitetura é usada corretamente.

Esse item levanta uma preocupação, ao passo que todo o esforço de definir, documentar e especificar uma arquitetura de software pode ser em vão, uma vez que essa arquitetura de software não seja utilizada da forma para a qual ela foi planejada. Cada projeto tem a sua especificidade, sendo que há casos em que essa certificação é tratada com baixa significância, como relatado por um dos participantes: "*A arquitetura dos projetos no modelo ágil tem uma "pegada" mais emergente e evolutiva e questões como validação se a arquitetura está sendo seguida, é quase que desnecessária, pois o arquiteto é parte do time e faz parte da entrega, já em métodos mais tradicionais (waterfall) ainda se pensa em desenhar o todo, como um grande esforço para se criar um blueprint/big picture e posteriormente passar o "bastão" para outras equipes de engenharia e desenvolvimento.*".

Ainda seguindo essa linha de raciocínio envolvendo o uso da arquitetura de software na prática, na questão "Eu me certifico que a modelagem está sendo feita de forma correta, para garantir que atributos de qualidade serão atendidos" (questão 9), o resultado é mais negativo, 29,5% dos entrevistados escolheram respostas negativas e 70,50% escolheram entre sempre ou muitas vezes. No entanto, de acordo com Tofan et al. (2014), em suas atividades, os arquitetos de software precisam considerar os requisitos funcionais e os atributos de qualidade (ou requisitos não funcionais) dos sistemas de software. Além disso, os atributos de qualidade desempenham um papel importante no processo de tomada de decisão, uma vez que os arquitetos de software podem comprometer alguns atributos de qualidade por meio de *tradeoffs* (por exemplo, segurança

versus usabilidade) (TOFAN et al., 2014).

Um exemplo semelhante ocorre quando o assunto é a definição e a documentação da arquitetura. Considerando a questão 1 "Eu defino a arquitetura de software", 80,60% dos respondentes escolheram as alternativas sempre ou muitas vezes, este é um grande número de respostas positivas, mas considerando o conteúdo e a importância da questão, uma vez que essa atividade é uma das principais atividades do arquiteto de software e é bem reconhecida por vários autores (KRUCHTEN, 2008; ERDER; PUREUR, 2016; SEI, 2017a), esse percentual deveria ser mais expressivo, ao invés de só ocupar a décima posição no ranking de questões com os maiores percentuais de respostas positivas, conforme apresentado na Tabela 3.1.

Porém, por mais que a atividade de definição da arquitetura de software seja uma das que sejam primeiro lembradas quando o assunto é a arquitetura de software em si, por meio desse comentário de um dos participantes é possível notar que nenhuma atividade pode ser tida como regra em todos os casos: "*Com meu trabalho atual não posso definir sempre questões arquiteturais, pois o cliente que trabalhamos tem sua equipe de arquitetura interna e eles que fazem a definição de qual arquitetura deverá ser utilizada. Porém para outros projetos sempre defino a arquitetura dos projetos utilizando o mais recente no mercado e participo de todas as fases do projeto, inclusive treinamento e acompanhamento [...]*".

Em contraste com o número de profissionais que responderam positivamente à questão 1, o mesmo não é verdade para a questão 2, "Eu crio documentos que descrevem a arquitetura de software", mesmo que seja de conhecimento geral que decisões documentadas facilitem a compreensão geral de um software, compartilhamento de conhecimento arquitetônico e processos de avaliação (KRUCHTEN; CAPILLA; DUEÑAS, 2009) e que a falta de documentação e compartilhamento do conhecimento arquitetural configure o antipadrão do arquiteto solitário (HOORN et al., 2011).

Um resultado preocupante é o percentual de respostas negativas referentes a documentação da arquitetura, 45,2% dos entrevistados responderam negativamente para a questão 2 e 54,80% escolheram alternativas sempre ou muitas vezes. Esse percentual coloca a questão 2 no ranking de questões com maior quantidade de respostas negativas, conforme apresentado na Tabela 3.2. A adoção de metodologias ágeis com maior foco em entregas e codificação e em detrimento da documentação pode ser uma das justificativas para isso ter acontecido.

Nessa observação pessoal de um dos participantes ele cita momentos em que a atividade de documentar acaba consumindo muito tempo e prejudicando sua participação de atividades mais técnicas: "*[...] acredito que é importante sim o papel do Arquiteto em desenhar e documentar soluções, mas participar do projeto tecnicamente é crucial na escolha das tecnologias e soluções. Digo isso pois vejo muitos arquitetos de software que muitas vezes nem sabem como está o código do projeto, o que muitas vezes não condiz com a solução proposta pelo mesmo*".

Uma outra possível implicação é que arquitetos de software aparentemente trabalham

isolados. Mais da metade (51,8%) dos entrevistados responderam negativamente sobre gestão de indivíduos e equipes, (questão 24), essa alta porcentagem levou a questão a ocupar a segunda posição do ranking de questões com maior índice de respostas negativas. Um dos antipadrões mais conhecidos é sobre arquitetos que trabalham isoladamente, os "arquitetos na torre de marfim".

A falta de trabalho em grupo pode ter influenciado a alta taxa de respostas negativas em questões como "Eu resolvo disputas e analiso *tradeoffs*" (questão 14), com 35,9%, e "Eu mantenho e elevo a moral da equipe responsável pela arquitetura" (questão 16), com 32,5%. Considerando que exista essa separação entre o arquiteto de software e a equipe que desenvolverá de fato o software, é possível que o antipadrão "criando uma arquitetura perfeita, mas muito difícil de implementar" seja colocado em prática.

Ainda em relação a questão sobre resolução de disputas e *tradeoffs*, há de se considerar que existem casos em que o arquiteto de software não possui autonomia para tal. Por exemplo, nessa assertiva feita por um dos participantes é possível entender a dinâmica: "*Vejo que é muito comum ao arquiteto atuar inicialmente em um papel consultivo, ou seja, demonstrando diferentes soluções para um problema, apresentando as vantagens e desvantagens de cada uma das soluções, porém algumas vezes o gerente/lider/cliente prefere tomar a decisão de qual solução utilizar. Um exemplo classico seria o de 2 soluções, uma que aumenta o custo da manutenção e diminui o desenvolvimento e outra que faz o oposto. Um cliente pode preferir uma manutenção mais barata visando um ciclo de vida maior, já um gerente pode preferir um custo de desenvolvimento melhor, pensando em uma entrega mais rápida.*".

Um fato que promove reflexão é que grande parte das questões com as maiores taxas de respostas negativas, apresentadas na Tabela 3.2, é composta de tarefas que exigem que o arquiteto de software esteja envolvido com a organização ou com outros *stakeholders*. Apesar dessa similaridade entre as questões esse fato não pode ser considerado uma regra, existem questões que também envolvem interação, como a questão 3, "Eu atuo como comunicador da arquitetura de software" e a questão 11, "Eu identifico e interajo com os stakeholders para certificar que suas necessidades serão atendidas", que receberam porcentagens consideráveis de respostas positivas, 77,6% e 72, 20% respectivamente.

Na questão 23, "Eu participo durante o planejamento dos projetos", cerca de 30% dos arquitetos de software optaram por respostas negativas, o que significa que esses profissionais não estão envolvidos nas etapas iniciais de definição do projeto de software. Essas duas observações feitas pelos participantes levantam outro ponto de reflexão, a relação entre empresa e arquitetura: "*[...] o papel do Arquiteto de Software muda demais com a fase que cada empresa está passando, por exemplo em empresas grandes o papel do Arquiteto é mais planejamento de construção de blueprints do que mão na massa, já em startups temos que dividir o tempo [...]*" e "*[...] Mas acredito que a valorização da arquitetura está iniciando no Brasil, aos poucos as empresas começam a perceber a vantagem de iniciar projetos com calma e bem planejados, pois isso acaba*

*custando menos e garantindo uma maior facilidade tanto em manutenção quanto suporte*".

Os arquitetos de software estão ainda menos envolvidos quando se trata de negócios e marketing, a maior parte deles nunca se envolveu nessas etapas. Como apresentado na questão 29, "Eu auxilio no marketing e futuras definições de produtos", com 82,3% de respostas negativas, ocupando a primeira posição no ranking de respostas negativas. Nenhum participante chegou a comentar especificamente sobre essas etapas, corroborando com a ideia de que a participação dos arquitetos de software nessa fase realmente é algo raro.

A questão 28, "Eu estou envolvido em todas as etapas do ciclo de vida do software" recebeu 38,7% de respostas negativas. Visto que a arquitetura de software é o elemento central em todo o ciclo de vida do software (WEINREICH; BUCHGEHER, 2012), é imperativo que o arquiteto de software tenha uma participação mais ativa em todas as fases do ciclo de vida. Porém não é em todos os projetos que essa participação ativa será uma regra, tem-se como exemplo essa afirmação feita por um dos participantes: "*Dependendo da necessidade do projeto a atuação/responsabilidades mudam um pouco. Em alguns casos apenas defino a arquitetura inicial e a evolução é acompanhada por outra pessoas, em outros o cliente já tem uma arquitetura de referência que deve ser apenas adaptada às necessidades do projeto e em outros cuido de todo o ciclo de desenvolvimento*".

Vale a pena mencionar duas outras questões sobre o envolvimento do arquiteto com a organização. A primeira é que 40,6% dos entrevistados responderam negativamente a "Eu me certifico que a administração entende os detalhes necessários da arquitetura" (questão 8), e a segunda é que 36,3% responderam negativamente a "Eu convenço os *stakeholders* sobre o valor da minha solução arquitetural" (questão 12). Esta comunicação ineficiente com os *stakeholders* pode levar a problemas de compreensão e aceitação da arquitetura de software proposta. Pode ser que essa falta de comunicação tenha influenciado a quantidade de respostas negativas (41,2%) recebidas pela pergunta " Eu me certifico que a arquitetura seja definida em etapas para que haja progresso na organização antes que a arquitetura seja completamente definida" (questão 6). Sem transparência e bom relacionamento torna-se muito difícil que a arquitetura de software seja definida de maneira eficiente.

Os arquitetos de software operam em um ambiente social e organizacional com diferentes *stakeholders* e forças que influenciam as tarefas que eles executam (GALSTER; TAMBURRI; KAZMAN, 2017). Um arquiteto de software que não está se comunicando tem uma probabilidade maior de não entender os propósitos do software. Os arquitetos de software devem garantir uma arquitetura de software sustentável e coerente com esses propósitos, logo, suas habilidades de comunicação e colaboração se tornam cada vez mais relevantes (ERDER; PUREUR, 2016).

Quanto às questões sobre conhecimento e habilidades técnicas, 73,90% responderam que sempre ou muitas vezes atuam como consultores da tecnologia (questão 27) e 83,4% mencionaram que sempre ou muitas vezes auxiliam em questões como seleção de ambientes e ferramentas (questão 10). A questão sobre resolução de problemas técnicos (questão 15) foi

muito semelhante (84,70%), sendo uma das questões com maior índice de respostas positivas.

Nota-se a importância da habilidade técnica nesse comentário de um participante: "*O Papel mais importante do Arquiteto de software é transformar os requisitos de negócio em ideias técnicas. Nem sempre é necessário que o arquiteto programe, mas principalmente é necessário que seja a ponte entre a gestão de produto e a equipe técnica*".

Em relação à questão subjetiva, podem ser vistas a seguir mais sentenças significativas escritas pelos participantes do estudo que ajudam a entender a realidade da profissão, mas não se encaixam especificamente no contexto de uma única questão:

"*No Brasil, na maioria das empresas de softwares, o papel do arquiteto não é bem definido, assim sendo, ele precisa atuar em todos os processos de desenvolvimento e até mesmo desenvolver muita vezes tudo que arquitetou se tornando um modelo de desenvolvedor que chamamos de full stack, comprometendo prazos e qualidade do software, logo, o arquiteto está sempre com o gerente de projetos e equipes*".

" *[...] na maioria das empresas, não é sempre que o arquiteto tem voz ativa e/ou poder de decisão. Infelizmente as decisões são tomadas com base no tempo/entrega, ou seja, não importa quanto custa ou se vai funcionar, o legal é implantar no prazo e mostrar para o stakeholder que está lá*".

"*Adoraria responder Sempre em todas as questões, mas em ambiente corporativo nem sempre a arquitetura tem o espaço necessário, muitas variáveis como comercial, ambiente legado, clientes com ambientes hostis e dificuldade de encontrar profissionais qualificados para execução (o que inviabiliza muitas vezes a adoção de novas tecnologias por questões de custo e curva de aprendizado), fazem com que o trabalho de arquiteto tenha uma dificuldade maior em sua execução [...]*".

Depois da análise das respostas do questionário, percebe-se que o conjunto de atividades encontradas teve, em sua maioria, aceitação de acordo com a vivência dos arquitetos de software na indústria de desenvolvimento de software. No entanto, nenhuma atividade apresentou unanimidade entre os profissionais, nem mesmo as que obtiveram as maiores porcentagens de respostas positivas.

## 3.5 Ameaças à Validade

Diferentes problemas podem ser ocasionados durante a participação dos indivíduos no questionário. Neste estudo, foram abordadas as ameaças internas, externas, de construção e conclusão:

(a) **Validade de conclusão**: Ao explicitar todas as trinta e seis perguntas utilizadas no questionário e todo o protocolo seguido para encontrar os arquitetos de software no LinkedIn e

outras plataformas, e a forma de envio dos questionários a serem respondidos, acredita-se que os resultados são válidos e podem ser replicados em estudos realizados em outros países.

(b) **Validade de construção**: No estudo de revisão sobre os papéis, habilidades e responsabilidades do arquiteto de software, houveram esforços para reunir o máximo de informações possível, como pode ser visto na Seção 3.1. Assim, acredita-se que os padrões encontrados representam vastamente os padrões ditados pela literatura na arquitetura de software.

(c) **Validade externa**: Apesar da pesquisa ter sido respondida apenas por profissionais do LinkedIn que residem no Brasil, é importante notar que parte desses profissionais podem trabalhar em empresas multinacionais ou remotamente em empresas estrangeiras, ter tido experiência acadêmica internacional (MBAs, mestrado ou doutorado no exterior), ou mesmo experiência de trabalhos anteriores em outros países. Tais experiências podem enviesar o resultado. O número de entrevistados é considerável quando comparado a outros levantamentos em pesquisa de Engenharia de Software, como por exemplo, 449 em (HUTCHINSON; WHITTLE; ROUNCEFIELD, 2014), 258 em (BESKER; MARTINI; BOSCH, 2017), 268 em (RAHMAN et al., 2017) e 99 em (RAUNAK; BINKLEY, 2017).

(d) **Validade interna**: Um erro típico é o mau uso da análise estatística (EASTERBROOK et al., 2008). Neste estudo, foram utilizadas estatísticas básicas para analisar dados, portanto, as ameaças de validade interna são mínimas.

# 4 Proposta de uma Disciplina de Arquitetura de Software em nível de Graduação

Neste Capítulo são discutidas a motivação e construção de uma proposta de disciplina de arquitetura de software em nível de graduação para instituições de ensino superior no Brasil. Na Seção 4.1 é apresentado um estudo sobre o panorama atual da arquitetura de software como disciplina isolada nas instituições de ensino superior no Brasil. Na sequência, a Seção 4.2 apresenta a versão inicial da proposta de plano de ensino e avaliação da mesma feita por professores de engenharia/arquitetura de software que lecionam no Brasil. Por fim, na Seção 4.3, são apresentados os tópicos referentes ao resultado final da proposta de plano de ensino após alterações sugeridas pelos professor em em suas avaliações.

## 4.1 Ensino da Arquitetura de Software no Brasil

Nos capítulos anteriores foi possível perceber a multiplicidade do papel do arquiteto de software, e principalmente após a aplicação do questionário e a análise das respostas obtidas, especialmente as respostas subjetivas, foi possível identificar dificuldades e divergências entre os participantes e seus empregadores em definir ou utilizar na prática os conceitos da arquitetura de software.

Com o objetivo de mapear o cenário atual do ensino da arquitetura de software em nível de graduação no Brasil, foi realizada uma busca nas páginas das instituições de ensino superior (IES) públicas que ofertam cursos de Sistemas de Informação (SI), Ciência da Computação (CC), Engenharia de Software (ES) e Engenharia da Computação (EC), com o intuito de descobrir quantas e quais instituições possuem a disciplina de Arquitetura de Software dentro da grade obrigatória desses cursos.

A ferramenta de busca utilizada para encontrar em totalidade as instituições públicas brasileiras de ensino superior foi a consulta avançada do site do Cadastro e-MEC (https://emec.mec.gov.br), site responsável pelo Cadastro Nacional de Cursos e Instituições de Educação Superior.

Os filtros de busca preenchidos na plataforma foram o curso de graduação (Sistemas de Informação, Ciência da Computação, Engenharia de Software e Engenharia da Computação), no campo Grau foi escolhida a opção “Bacharelado”, a opção “Sim” para a Gratuidade do curso e a alternativa “Em Atividade” para o campo Situação. Foram realizadas quatro buscas, uma para

cada curso. A tela de consulta do Cadastro e-Mec com a totalidade dos filtros pode ser vista na Figura 4.1.

As buscas, realizadas entre os dias 20 de agosto e 12 de setembro de 2018, retornaram 127 cursos de CC, 98 cursos de SI, 15 cursos de ES e 42 cursos de EC. Para encontrar a grade curricular dos cursos foi necessário acessar manualmente o site de cada uma das instituições de ensino correspondentes, localizar a página do respectivo curso e identificar se a disciplina de arquitetura de software estava presente dentre as disciplinas obrigatórias da grade curricular em questão.

Uma vez que a pesquisa foi realizada manualmente em cada uma das páginas das IES, a gratuidade do curso foi um critério escolhido como filtro na pesquisa, pois caso os cursos de IES privadas também fossem contemplados, o total de cursos pesquisados seria 1197 (sendo 424 cursos de CC, 536 de SI, 118 de EC e 90 de ES), dificultando a execução da pesquisa.

Figura 4.1 – Tela consulta avançada Cadastro e-Mec.


Fonte: Cadastro e-Mec (2019).

Em relação aos cursos de CC e SI, foram encontrados respectivamente 2 e 3 cursos que possuem a disciplina Arquitetura de Software como disciplina obrigatória dentro da sua grade, ou seja, apenas 1,57% e 3,06%. A respeito do curso de ES o cenário é mais positivo, dos 15 cursos encontrados na busca 9 deles possuem a disciplina, ou seja 60%. No tocante ao curso de Engenharia da computação nenhum dos 42 cursos encontrados possui a disciplina. Os 14 cursos encontrados no total, somando-se as 4 buscas, podem ser vistos na Tabela 4.1, composta pela sigla, IES, campus, curso e ano da grade curricular.

Nas últimas décadas, a arquitetura de software emergiu como um subcampo importante da engenharia de software. Durante esse período, houve um progresso no desenvolvimento

da base tecnológica e metodológica para o tratamento da arquitetura de software como uma disciplina de engenharia (GARLAN, 2014). No entanto, ainda há muito para ser feito quanto a arquitetura de software como disciplina isolada. Pode-se notar que hoje, principalmente no tocante aos cursos de CC e SI, a porcentagem de cursos que ensinam isoladamente arquitetura de software na graduação é muito pequena em comparação com a totalidade dos cursos.

Antonino, Morgenstern e Kuhn (2016), ao estudarem sobre os arquitetos de software chegaram a conclusão que existe a necessidade de abordar a arquitetura de software na educação formal. Segundo os autores, os cursos de ciência da computação devem possuir mais ênfase na disciplina de arquitetura, os cientistas devem pensar além do código-fonte, devem ser ensinados a arquitetar melhor os softwares, não ensinar apenas padrões de design, apesar de ser um tópico fundamental, mas também incluir na educação dos alunos o papel do arquiteto de software e projeto e avaliação de arquitetura de alto nível.

De acordo com os resultados obtidos por meio do *survey* realizado nesta pesquisa, Seção 3.4, percebe-se que o cenário no Brasil não é diferente. Existe uma lacuna entre o ensino reduzido sobre arquitetura de software ofertado aos alunos em nível de graduação e o que é cobrado ao arquiteto de software como profissional no mercado de trabalho. Logo, como forma de mitigar essa lacuna, foi detectada a necessidade de propor um plano de ensino de arquitetura de software que pudesse ser adicionado à grade desses cursos.

Quadro 4.1 – Cursos com Arquitetura de Software obrigatória na grade curricular.

| SIGLA | IES | CAMPUS | CURSO | ANO GRADE |
|---|---|---|---|---|
| USP | Universidade de São Paulo | São Carlos | CC | 2014 |
| UEM | Universidade Estadual de Maringá | Maringá | CC | 2016 |
| UEG | Universidade Estadual de Goiás | Santa Helena de Goiás | SI | 2019 |
| IFGoiano | Instituto Federal de Educação, Ciência e Tecnologia Goiano | Urutaí | SI | 2015 |
| IFNMG | Instituto Federal de Educação, Ciência e Tecnologia do Norte de Minas Gerais | Januária | SI | 2016 |
| UNB | Universidade de Brasília | Gama | ES | 2017 |
| UFRN | Universidade Federal do Rio Grande do Norte | Natal | ES | 2014 |
| UDESC | Fundação Universidade do Estado de Santa Catarina | Ibirama | ES | 2017 |
| UFC | Universidade Federal do Ceará | Russas | ES | 2018 |
| UFG | Universidade Federal de Goiás | Goiânia | ES | 2017 |
| UTFPR | Universidade Tecnológica Federal do Paraná | Cornélio Procopio | ES | 2018 |
| UTFPR | Universidade Tecnológica Federal do Paraná | Dois Vizinhos | ES | 2018 |
| UFMS | Universidade Federal do Mato Grosso do Sul | Campo Grande | ES | 2018 |
| UFERSA | Universidade Federal Rural do Semi-Árido | Pau dos Ferros | ES | 2018 |

## 4.2 Proposta e Avaliação de Plano de Ensino

Segundo Baffi (2002), planejar, em sentido amplo, é um processo que "visa a dar respostas a um problema, estabelecendo fins e meios que apontem para sua superação, de modo a atingir objetivos antes previstos, pensando e prevendo necessariamente o futuro". Planejar, dentro da educação, tem como características básicas evitar a improvisação, prever o futuro, estabelecer caminhos que possam nortear mais apropriadamente a execução da ação educativa, prever o acompanhamento e a avaliação da própria ação.

O plano, fruto do planejamento, é um documento utilizado para o registro de decisões que envolvem o que se pensa fazer, como fazer, quando fazer, com o que fazer e com quem fazer. Para existir plano é necessária a discussão sobre fins e objetivos, culminando com a definição dos mesmos, pois somente desse modo é que se pode responder a essas questões.

Em posse dessas definições, foi criada uma proposta de plano de ensino da arquitetura de software para instituições de ensino superior no país. A proposta subdivide-se em: Ementa, Objetivos, Conteúdo Programático, Metodologia e Forma de Avaliação e Bibliografia.

O documento contendo a versão inicial da proposta de plano de ensino, que pode ser visto no Apêndice D, foi enviado por e-mail para 42 professores de engenharia de software e/ou arquitetura de software com o intuito de avaliar a corretude e aplicabilidade do plano de ensino proposto. O texto do e-mail enviado pode ser visto no Apêndice C. Os professores foram selecionados por meio de pesquisa manual em instituições de ensino superior localizadas em diversos estados do país que disponibilizam acesso público ao seu corpo docente.

Sete avaliações foram recebidas, a primeira delas de um professor que atualmente ministra no mestrado profissional da Universidade de Brasília a disciplina Arquitetura de Software. A ementa e todo o material do curso foram criados pelo mesmo. Além disso o professor é co-autor do livro "Documenting Software Architectures: Views and Beyond" escrito em parceria com autores como Paul Clements e David Garlan. Sua avaliação sugeriu alterações principalmente na bibliografia proposta.

Inicialmente a proposta continha em sua bibliografia complementar os livros:

1. BOOCH, Grady. Handbook of Software Architecture (disponível na web).
2. CLEMENTS, P.; KAZMAN, R.; KLEIN, M. H. Evaluating Software Architectures: Methods and Case Studies, Addison-Wesley, 2001.

Os dois foram retirados sob justificativa que o primeiro faz parte de um projeto descontinuado e o segundo é antigo e pode ser suprimido pois o livro *"Software Architecture In Pratice"*, presente na bibliografia, possui uma boa seção sobre avaliação. Com a remoção dos livros a bibliografia tornou-se mais enxuta e objetiva. Essa avaliação ainda considerou o conteúdo do livro *"Software Architecture: Foundations, Theory, and Practice"* muito teórico e o livro

*"Pattern-oriented Software Architecture, on Patterns and Pattern Languages"* suprido pelos livros e leituras mais modernas também presentes na bibliografia, porém por ambos livros serem clássicos foi decidido que não seriam retirados.

Houve também sugestão de alteração no texto da forma de avaliação, que inicialmente considerava o projeto subdividido em duas partes, como 20% das avaliações individuais 1 e 2, tornando a compreensão de como a nota do projeto seria mensurada mais difícil. Ao adicionar o projeto como 40% da nota final houve um melhor entendimento da participação do projeto na avaliação.

A segunda avaliação, realizada por um professor Doutor em Ciência da Computação pela Universidade de São Paulo, que atualmente ministra a disciplina de Engenharia de Software no Departamento de Informática da Universidade Estadual de Maringá (DIN-UEM), também contribuiu para a pesquisa apontando melhorias como a adição de mais recursos didáticos e um fórum para comunicação com os alunos fora da sala de aula na seção de metodologia. Comentários similares a avaliação anterior sobre a participação do projeto na seção Forma de Avaliação também foram feitos.

Foram suas as sugestões de inclusão do livro *"Software Metrics: A Rigorous and Practical Approach"* e retirada do livro *"Object-oriented metrics in practice: using software metrics to characterize, evaluate, and improve the design of object-oriented systems"* na bibliografia complementar e da inclusão de "Noções sobre arquitetura de referência" e "Arquitetura de linha de produto" na ementa.

A terceira avaliação, realizada por um professor que ministra disciplinas de Engenharia de Software, Algoritmos e lógica de programação, Sistemas de apoio a decisão, entre outras, na Universidade Federal do Rio Grande do Norte sugeriu a retirada do item "Padrões de projeto" dentre os itens presentes na ementa.

A quarta avaliação, realizada por um professor Doutor que ministra a disciplina de engenharia de software na Universidade de Uberlândia, elogiou a proposta do ponto de vista teórico/conceitual e sugeriu adicionar um tema mais prático e atual como a arquitetura de *microsservices*.

A quinta avaliação, realizada por um professor Doutor que ministra atualmente disciplinas como Princípios de Engenharia de Software, Análise e Projeto de Sistemas, Qualidade de Software, entre outras na Universidade Federal Rural do Semi-Árido sugeriu, similar à primeira e segunda avaliação, alteração no cálculo da forma de avaliação para que ficasse mais clara a participação do projeto na média final.

As outras duas avaliações recebidas continham apenas aprovação e aceite, sem adição de observações ou sugestões. A versão final da proposta de plano de ensino contendo as alterações sugeridas na fase de avaliação pode ser vista na próxima seção.

## 4.3 Versão Final da Proposta de Plano de Ensino

Nesta Seção são discutidos os tópicos referentes a versão final da proposta de plano de ensino da disciplina de arquitetura de software a nível de graduação. Nas subseções a seguir serão discutidas a Ementa, Objetivos, Conteúdo Programático, Metodologia e Forma de Avaliação e Bibliografia.

### 4.3.1 Ementa

A ementa do curso é um texto curto e resumido que contém o essencial. Pode ser tratada como um resumo ou sinopse de determinado tema ou área. Os itens inclusos na ementa são:

- Introdução à arquitetura de software – conceito, objetivos e escopo;
- Importância e impacto da Arquitetura de software;
- Processo para a construção de arquitetura de software;
- Elementos básicos de uma arquitetura de software;
- O papel do arquiteto de software no ciclo de desenvolvimento;
- Estilos arquiteturais;
- Notações arquiteturais;
- Noções sobre arquitetura de referência;
- Arquiteturas específicas de domínio e *Frameworks*;
- Arquitetura de linha de produto;
- Arquitetura de *microsservices*;
- Modelagem de Arquitetura de Software;
- Linguagens de descrição de arquitetura (ADL);
- Métodos para a avaliação de arquiteturas de software;
- Ferramentas de suporte à construção, representação e avaliação de arquiteturas de software;
- Gerenciamento de riscos arquiteturais; e,
- Reutilização de componentes arquiteturais.

Durante a busca nas páginas das IES pela existência da disciplina de Arquitetura de software na grade dos cursos, onze ementas de Arquitetura de software foram encontradas. Essas onze ementas serviram como modelo base para a versão inicial da ementa proposta. Porém, para suprir a necessidade em relação ao que foi exposto nos resultados do capítulo anterior e buscando alcançar a totalidade de assuntos necessários, foram adicionados os tópicos: O papel do arquiteto de software no ciclo de desenvolvimento, Reutilização de componentes arquiteturais, Ferramentas de suporte à construção e Representação e avaliação de arquiteturas de software. Esses tópicos buscam cobrir as lacunas relacionadas diretamente ou indiretamente às atividades que apresentaram altos índices negativos de respostas dos entrevistados no *survey*. Os tópicos Noções sobre arquitetura de referência, Arquitetura de linha de produto e Arquitetura de *microsservices* foram adicionados a essa versão final após sugestão na fase de avaliação.

### 4.3.2 Objetivos

O objetivo geral dessa proposta de plano de ensino subdivide-se em oferecer aos alunos uma visão abrangente de arquitetura de software apresentando métodos e técnicas para representar e avaliar arquiteturas, discutir o impacto da arquitetura de software dentro do ciclo de vida de desenvolvimento do software, bem como o papel do arquiteto de software no mesmo, oferecer uma visão sobre o uso de ferramentas de apoio à construção, representação e avaliação de arquiteturas, e discutir as formas de mitigação de riscos e reutilização de arquiteturas.

De forma a alcançar o objetivo geral, os seguintes objetivos específicos foram definidos:

- Aplicar conceitos de arquitetura de software para definir, documentar, comunicar e avaliar uma arquitetura de software;
- Representar a arquitetura e o *design* de um software e seus componentes utilizando uma linguagem de descrição de arquitetura;
- Desenvolver em equipe uma arquitetura executável (versão inicial de um software), colocando em prática os conceitos vistos em sala.

Com esses objetivos busca-se uma forma de ensino que alinhe a teoria e a prática, formando um profissional mais preparado para os desafios da profissão. Com as atividades em grupo o intuito é que o aluno possa desenvolver habilidades de comunicação e conciliação com os demais *stakeholders* do projeto. A respeito dos objetivos não houveram sugestões de alteração entre a versão inicial e a versão final.

### 4.3.3 Conteúdo Programático

O conteúdo programático é o roteiro a ser seguido pelo instrutor no decorrer da disciplina. Ele organiza em que sequência e quando os tópicos referentes à ementa serão apresentados. Para

esse plano de ensino cada aula deve possuir uma duração de 2 horas. O conteúdo programático pode ser visto a seguir:

- AULA 01: Introdução. Plano de ensino da disciplina. Explicação sobre o projeto.
- AULA 02: Conceitos e objetivos da arquitetura de software. Importância e impacto em um software.
- AULA 03: O papel do Arquiteto de Software no ciclo de desenvolvimento de software.
- AULAS 04 e 05: Processos para a construção de arquitetura de software. Elementos básicos de uma arquitetura de software.
- AULAS 06 e 07: Estilos arquiteturais (*pipe-and-filter*, *multi-tier*, *layered*, transações, *publish-subscribe*, baseado em eventos, cliente-servidor, MVC e outros).
- AULA 08. Norma ISO/IEC/IEEE 42010 (*concerns*, *stakeholders*, visões, *rationale*, modelo arquitetural, entre outros).
- AULAS 09 e 10: Aulas práticas.
- AULAS 11 e 12: Avaliação 1.
- AULAS 13 e 14: Notações arquiteturais (visões, representações, diagramas de componentes e outros). Exercícios.
- AULAS 15 e 16: Noções sobre arquitetura de referência. Arquiteturas específicas de domínio e *Frameworks*. Arquitetura de linha de produto. Arquitetura de *Microsservices*.
- AULAS 17 e 18: Modelagem: UML, MDA. Exercícios.
- AULA 19: Métodos para a avaliação de arquiteturas de software
- AULAS 20 e 21: Linguagens de descrição de arquitetura (ADL). Ferramenta de suporte à construção, representação e avaliação de arquiteturas de software.
- AULAS 22 e 23: Gerenciamento de riscos arquiteturais. Reutilização.
- AULAS 24 e 25: Aulas práticas.
- AULAS 26 e 27: Avaliação 2.
- AULA 28: Devolução de provas.
- AULAS 29 e 30: Seminários do projeto.

As aulas práticas serão momentos em que dúvidas e orientações sobre o projeto poderão ser discutidas e oferece um espaço para que o aluno possa, em equipe, tomar decisões e implementar de fato a arquitetura proposta. As aulas marcadas como avaliação são horários reservados para a resolução de avaliações teóricas sobre o assunto decorrido até aquele momento. As demais aulas representam com um maior nível de detalhamento uma direção de como os tópicos vistos na ementa devem ser expostos. A versão final do conteúdo programático foi alterada para comportar os novos tópicos adicionados na ementa por meio das alterações sugeridas na avaliação.

### 4.3.4 Metodologia e Forma de Avaliação

A metodologia visa aproximar e preparar o aluno para situações que podem ocorrer durante o exercício da profissão, gerando maior segurança e autonomia ao serem expostos ao mercado de trabalho. Também é possível verificar por meio da metodologia se os objetivos foram alcançados, se os alunos consolidaram a aprendizagem e se a atuação docente foi adequada quanto aos objetivos, conteúdos, metodologia, relacionamento professor/aluno, procedimentos de avaliação e duração das aulas.

A fim de propiciar uma aprendizagem efetiva por parte dos alunos, o plano de ensino propõe que o conteúdo seja trabalhado a partir de aulas expositivas, resolução de exercícios em sala de aula, de modo que a participação seja incentivada, e desenvolvimento de trabalhos práticos. Nas aulas práticas o grupo deve apresentar os artefatos do projeto de acordo com o planejamento inicial que será apresentado na primeira aula da disciplina.

Como apoio extra sala de aula, os alunos poderão tirar suas dúvidas com o professor em horário definido pelo mesmo ou postar suas dúvidas no fórum da turma. As aulas serão ministradas com o apoio de recursos didáticos variados, como dispositivos aúdio-visuais, quadro, lousa, giz, entre outros.

No tocante à forma de avaliação, esta será composta por duas provas teóricas e exercícios realizados em sala de aula e o projeto desenvolvido durante a disciplina, como segue:

- Avaliação 1: 1 prova teórica, individual, sem consulta
- Avaliação 2: 1 prova teórica, individual, sem consulta

Na versão inicial as notas de cada uma das unidades era composta por 60% de nota de prova e 40% de nota do projeto. Porém, na formulação da versão final, após sugestões de alteração na fase de avaliação, a nota de cada uma das unidades será composta de 80% de nota da avaliação e 20% de nota de exercícios feitos em sala de aula. A nota final será composta por 60% referente à media das avaliações e 40% de nota do projeto (NP).

- Nota final = ((A1+A2)/2)*0.6 + NP*0.4

O projeto da disciplina será feito em grupo, porém avaliado de maneira individual, de forma a incentivar que todos os alunos participem efetivamente do projeto. As datas de avaliação

deverão ser divulgadas no primeiro dia de aula, assim como demais atividades da disciplina e como a avaliação ocorrerá.

Essa forma de avaliação foi escolhida para que o aluno permaneça constantemente motivado para a obtenção de sua aprovação. Dessa forma, não haverá predileção entre as atividades propostas, por exemplo, buscando aprovação apenas por meio das notas adquiridas nas provas em detrimento da pontuação do projeto. Mas sim, que ele possa se envolver em tudo que é ofertado na disciplina e se engaje o suficiente para que a soma dos seus esforços reflita o aprendizado durante o decorrer da disciplina.

### 4.3.5 Bibliografia

Para apoio a esse plano de ensino foram escolhidas bibliografias básicas e complementares, onde todo o conteúdo da ementa da disciplina pode ser encontrado.

Como bibliografia básica, foram escolhidos os seguintes livros:

1. BASS, L.; CLEMENTS, P.; KAZMAN, R. *Software Architecture In Practice*. Third Edition. Addison-wesley, 2013.
2. CLEMENTS, Paul et al. *Documenting Software Architectures: Views and Beyond*. Addison-Wesley, 2010.
3. ROZANSKI, Nick; WOODS, Eóin. *Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives.* Addison-Wesley, 2011.
4. BUSCHMANN, Frank; HENNEY, Kevin; SCHMIDT, Douglas C. *Pattern-oriented Software Architecture, on Patterns and Pattern Languages.* John wiley & sons, 2007.
5. TAYLOR, Richard N.; MEDVIDOVIC, N.; ERIC M. Dashofy. *Software Architecture: Foundations, Theory, and Practice.* Wiley, 2009.

Os cinco livros foram escolhidos buscando cobrir em totalidade os assuntos expostos na ementa e tratados durante a disciplina. Todos eles foram escritos por autores reconhecidos na área da Arquitetura de Software.

No primeiro livro, os autores com base em sua própria experiência abordam os tópicos técnicos essenciais para projetar, especificar e validar a arquitetura de um software. Eles também enfatizam a importância do contexto de negócios no qual os sistemas são projetados. Seu objetivo é apresentar a arquitetura de software em um cenário do mundo real, refletindo tanto as oportunidades quanto as restrições que as empresas enfrentam.

O segundo livro fornece uma orientação mais completa sobre como capturar uma arquitetura de uma forma comumente compreensível, independente de linguagem ou notação. Com

base em sua experiência, os autores auxiliam na decisão de quais informações documentar e, com diretrizes e exemplos (em várias notações, incluindo UML), apresentam como expressar uma arquitetura para que outras pessoas possam criar e utilizá-la com êxito.

Ainda no segundo livro são apresentadas regras para documentação segura, os objetivos e estratégias de documentação, modos de exibição de arquitetura e estilos, documentação para interfaces e comportamentos de software e modelos para capturar e organizar informações.

O terceiro livro é um guia voltado para profissionais sobre projeto e implementação de arquiteturas eficazes para sistemas de informação. É uma introdução à arquitetura de software e um manual de práticas recomendadas bem estabelecidas. Aborda temas como projeto e comunicação de arquitetura equilibrando as diferentes necessidades dos *stakeholders*, aspectos arquiteturalmente significativos do design, incluindo áreas como desempenho, resiliência e localização, desenvolvimento ágil e arquitetura de software trabalhando juntos, entre outros.

O quarto livro aborda a arquitetura de software orientada a padrões. Os padrões definem como o software é projetado, construído e documentado. O livro oferece uma visão aprofundada dos padrões existentes e como usá-los com sucesso, além de servir como um manual de referência para todas as arquiteturas de software orientadas a padrões.

O quinto livro aborda as faces da arquitetura de software e demonstra como ela é uma peça central do desenvolvimento e evolução do software. O livro detalha técnicas de modelagem, design, implementação, implantação e adaptação do sistema - além de diversos outros tópicos - colocando os elementos em contexto e comparando-os.

A respeito da bibliografia básica houveram sugestões de alterações na versão inicial que não foram acatadas na versão final da proposta, assim permanecendo inalterada. Porém, em relação a bibliografia complementar, houveram alterações a partir das sugestões na avaliação. A versão inicial da bibliografia complementar era composta por quatro livros, dos quais três foram removidos e mais um foi adicionado na versão final, tornando a bibliografia mais objetiva. Como bibliografia complementar, foram escolhidos os livros:

1. FENTON, Norman; BIEMAN, James. *Software Metrics: A Rigorous and Practical Approach.* Third Edition. CRC press, 2014.

2. SILVEIRA, Paulo; SILVEIRA, Guilherme; LOPES, Sérgio; MOREIRA, Guilherme; STEPPAT, Nico; KUNG, Fabio. Introdução à Arquitetura e Design de Software – Uma Visão Sobre a Plataforma Java. Rio de Janeiro: Elsevier – Campus, 2012.

O primeiro livro da bibliografia complementar fornece uma introdução abrangente às métricas de software e diretrizes práticas para selecionar métricas e planejar seu uso em um programa de medição. São discutidos assuntos como design orientado a objeto, padrões de design, desenvolvimento orientado a modelo e processos de desenvolvimento ágil, entre outros.

O segundo livro da bibliografia complementar oferece uma visão técnica e prática para elucidar muitas questões enfrentadas na utilização da plataforma Java. A proposta deste livro é apresentar a plataforma em nível de detalhamento, desde a descrição do bytecode e da JVM até a possibilidade de utilização de outras linguagens. Também são vistos conceitos de orientação a objetos, entre eles, herança, encapsulamento, uso de interfaces, *domain driven design*, injeção de dependências e más práticas.

# 5 Conclusão

O desenvolvimento do presente estudo possibilitou um melhor entendimento sobre o desempenho do papel do arquiteto de software no Brasil, e como ele se distancia ou se aproxima do que é encontrado na literatura internacional sobre a profissão.

O processo do *survey* foi descrito em detalhes. Inicialmente uma pesquisa foi realizada para encontrar os padrões e antipadrões a respeito da profissão do arquiteto de software descritos na literatura. Em seguida foi criado em questionário, instrumento de coleta de dados para o *survey*, no qual cada questão representa uma atividade derivada dos padrões encontrados. Esse formato possibilitou que cada padrão pudesse ser mensurado a partir do nível de desempenho daquela atividade dentro do exercício da profissão do arquiteto de software no seu cotidiano.

O processo de coleta de dados também foi descrito em detalhes. Para entrar em contato com o maior número possível de arquitetos de software no Brasil foi utilizada, majoritariamente, a plataforma LinkedIn, uma rede social de negócios voltada para contatos profissionais. Após realizar a busca dentro da plataforma por arquitetos de software que atuam no Brasil, foi necessário abrir cada perfil individual e manualmente para que a solicitação de amizade juntamente com a mensagem contendo convite para preenchimento do questionário fossem enviadas.

Além dos 3187 convites individuais enviados via LinkedIn, o questionário também foi enviado ao comitê de arquitetura de software de uma das maiores empresas multinacionais de software do Brasil (com quarenta e cinco membros no comitê arquitetural, e um total de cerca de quinze mil funcionários trabalhando no Brasil e vinte e cinco mil no mundo todo). O questionário obteve 536 respostas, contemplando as 72 respostas a questão aberta, descritiva e opcional.

Foi realizada a análise das respostas obtidas chegando a conclusão que em sua grande maioria os profissionais desempenham no seu cotidiano atividades que estão de acordo com os padrões encontrados na literatura, porém há algumas atividades específicas que ficaram bem abaixo da média de respostas das demais, por exemplo, “Eu auxilio no marketing e futuras definições de produtos”, “Eu gerencio indivíduos e equipes” e “Eu crio documentos que descrevem a arquitetura de software”, entre outros, configurando-se como pontos de atenção.

A partir da compilação da análise dos resultados, evidenciou-se a necessidade de investigar como ocorre atualmente o ensino da disciplina de arquitetura de software no Brasil, logo foi realizada uma busca pela disciplina nas grades curriculares dos cursos de SI, CC, ES e EC nas instituições públicas de ensino superior do Brasil que ofertam os mesmos. Nesse estudo foi possível notar que a porcentagem de cursos de SI e CC que possuem a disciplina como obrigatória em sua grade curricular é muito pequena frente à quantidade total de cursos. No tocante ao curso de EC nenhum dos cursos possui a disciplina. O cenário com ES é mais positivo, porém ainda contempla pouco mais da metade dos cursos.

Como forma de mitigar essa carência do ensino de arquitetura de software nos cursos investigados, foi criada uma proposta de disciplina de arquitetura de software para cursos de nível superior no Brasil.

Por fim, com o intuito de avaliar essa proposta de disciplina a mesma foi enviada para uma lista previamente reunida contendo 42 professores de engenharia/arquitetura de software que lecionam em diversas instituições de ensino superior do Brasil. Sete professores avaliaram a proposta e deram suas contribuições. A proposta foi então alterada para se adequar as sugestões e considerações enviadas.

## 5.1 Principais Contribuições

A principal contribuição desse trabalho foi a melhor compreensão sobre as características da profissão do arquiteto de software no Brasil a partir da análise das 536 respostas obtidas por meio de profissionais que atuam ou atuaram em algum momento de suas carreiras como arquitetos de software.

As informações fornecidas nesse trabalho podem ser utilizadas para definições e ações futuras por parte de instituições de ensino superior do país, conselhos de TI que venham a surgir, comunidades de arquitetura de software, empresas que desejam investir em treinamentos para os funcionários, ou até mesmo auxiliar empresas da área de recrutamento na busca por profissionais qualificados.

Além da compreensão sobre a profissão do arquiteto de software outros subprodutos desta dissertação merecem ser destacados, são eles, o quantitativo de cursos de SI, CC, ES e EC em instituições públicas de ensino superior do Brasil que possuem a disciplina de Arquitetura de Software em suas ementas como disciplina obrigatória e, consequentemente, a proposta de uma disciplina de Arquitetura de Software em nível de graduação.

Outra contribuição a ser destacada é o próprio referencial teórico que constitui uma compilação de definições e estudos sobre Arquitetura de Software e a profissão do arquiteto de software que pode ser utilizada para consultas futuras sobre os referidos temas.

É necessário destacar também a contribuição desse trabalho para a área de Arquitetura de Software visto que o mesmo teve seus resultados publicados em conferência internacional, sendo ela:

OLIVEIRA, Manoela R., VIEIRA, Felipe J.R., MISRA, Sanjay., SOARES, Michel Santos. **A Survey on the Skills, Activities and Role of the Software Architect in Brazil**. In: International Conference Computational Science and Its Applications – ICCSA 2019. Lecture Notes in Computer Science, vol 11623. Springer, Cham. p. 43-58. Qualis A4. São Petesburgo, Rússia, 2019.

## 5.2 Limitações e Trabalhos Futuros

Para a produção deste trabalho algumas limitações foram identificadas. A primeira delas refere-se à quantidade de respostas obtidas, apesar de terem sido contatados 3232 profissionais apenas 536 responderam ao questionário enviado.

Outra limitação do trabalho foi o instrumento de pesquisa do *survey* para coleta de informações possuir apenas uma questão aberta, impossibilitando que os entrevistados comentassem cada uma de suas respostas individualmente. Dessa forma pode-se ter deixado de coletar informações que vão além das respostas parametrizadas, porém devido a grande quantidade de questões presentes no questionário, adicionar uma questão aberta para cada questão objetiva faria com que o questionário se tornasse muito extenso e consequentemente cansativo para o preenchimento dos entrevistados.

Mais uma limitação do trabalho foi a pesquisa manual realizada nas páginas das IES do Brasil em busca das grades curriculares dos cursos de ES, SI, CC e EC, o que determinou que um número menor de instituições fizessem parte do estudo. Sendo assim, as instituições privadas não fizeram parte da pesquisa, bem como os cursos de pós-graduação *lato sensu* e *stricto sensu*.

Ainda como limitação do trabalho, vale destacar que a avaliação da proposta de uma disciplina de arquitetura de software foi feita por apenas sete professores de engenharia/arquitetura de software. Apesar do número de professores contatados ter sido maior, apenas sete participaram da avaliação.

Pretende-se para trabalhos futuros realizar uma análise estatística mais avançada, por exemplo, fazendo correlações com o tempo de trabalho do profissional e o padrão de resposta observado, a fim de descobrir se a experiência na profissão altera a sua atuação e percepção da mesma. Seguindo a mesma linha de raciocínio pode-se estudar o impacto da região do país em que o profissional atua.

Um fato que também merece menção em trabalhos futuros é o número relevante de profissionais que não tem nível superior mas atuam como arquitetos de software e a baixa porcentagem de doutores na área. Durante o trabalho atual foram levantadas hipóteses. Sobre os profissionais sem nível superior, pode ser reflexo da falta de regulamentação oficial, uma vez que ainda não existe um conselho ou regulamentação sobre o exercício das profissões de Tecnologia da Informação (TI) no Brasil, logo, pode ser um profissional com experiência na área mas que optou em não cursar um nível superior. Em relação a baixa porcentagem de doutores, pode ser justificado pelo fato da arquitetura ser uma área muito técnica e no Brasil a cultura do doutorado ser mais voltada para a área acadêmica, logo pode não ter se tornado uma opção de profissionalização atraente aos arquitetos de software.

Por fim, um outro trabalho futuro com grande relevância é a replicação do estudo com profissionais que atuam em outros países, com o intuito de comparar e identificar como a profissão é desempenhada em outras culturas.

# Referências


ANTONINO, P. O.; MORGENSTERN, A.; KUHN, T. Embedded-Software Architects: It's Not Only about the Software. *IEEE Software*, IEEE, v. 33, n. 6, p. 56–62, 2016.

BAFFI, M. A. T. O Planejamento em Educação: Revisando Conceitos Para Mudar Concepções e Práticas. *BELLO, José Luiz de Paiva. Pedagogia em Foco, Petropólis*, 2002.

BARROSO, A. S. et al. Influence of Human Personality in Software Engineering - A Systematic Literature Review. In: *ICEIS 2017 - Proceedings of the 19th International Conference on Enterprise Information Systems, Volume 3, Porto, Portugal, April 26-29, 2017*. [S.l.: s.n.], 2017. p. 53–62.

BASS; CLEMENTS; KAZMAN. *Software Architecture in Practice*. 3. ed. New Jersey, USA: Addison-Wesley, 2012.

BESKER, T.; MARTINI, A.; BOSCH, J. Impact of architectural technical debt on daily software development work—a survey of software practitioners. In: IEEE. *2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. [S.l.], 2017. p. 278–287.

BOOCH, G. The Economics Of Architecture-First. *IEEE Software*, IEEE, v. 24, n. 5, p. 18–20, 2007.

CHRISTENSEN, H. B.; HANSEN, K. M.; SCHOUGAARD, K. R. An empirical study of software architects' concerns. In: IEEE. *2009 16th Asia-Pacific Software Engineering Conference*. [S.l.], 2009. p. 111–118.

CLEMENTS, P. et al. The Duties, Skills, and Knowledge of Software Architects. In: IEEE. *Working IEEE/IFIP Conference on Software Architecture (WICSA'07)*. [S.l.], 2007. p. 20.

CORREA, B. How Are Architects Made? *IEEE software*, IEEE, v. 30, n. 5, p. 11–13, 2013.

EASTERBROOK, S. et al. Selecting Empirical Methods for Software Engineering Research. In: *Guide To Advanced Empirical Software Engineering*. [S.l.]: Springer, 2008. p. 285–311.

EELES, P.; CRIPPS, P. *The Process of Software Architecting*. Massachusetts, USA: Addison-Wesley Professional, 2010.

ERDER, M.; PUREUR, P. What's the Architect's Role in an Agile, Cloud-Centric World? *IEEE Software*, IEEE, v. 33, n. 5, p. 30–33, 2016.

FOWLER, M. Who Needs An Architect? *IEEE Software*, IEEE, n. 5, p. 11–13, 2003.

GALSTER, M.; TAMBURRI, D. A.; KAZMAN, R. Towards Understanding the Social and Organizational Dimensions of Software Architecting. *ACM SIGSOFT Software Engineering Notes*, ACM, v. 42, n. 3, p. 24–25, 2017.

GARLAN, D. Software Architecture: A Roadmap. In: CITESEER. *Proceedings of the Conference on the Future of Software Engineering*. [S.l.], 2000. p. 91–101.

GARLAN, D. Software Architecture: A Travelogue. In: ACM. *Proceedings of the on Future of Software Engineering*. New York, NY, USA, 2014. (FOSE 2014), p. 29–39.

GORTON, I. Understanding Software Architecture. In: *Essential Software Architecture*. [S.l.]: Springer, 2011. p. 1–15.

HOHPE, G. et al. The Software Architect's Role in the Digital Age. *IEEE Software*, IEEE, v. 33, n. 6, p. 30–39, 2016.

HOORN, J. F. et al. The Lonesome Architect. *The Journal of Systems and Software*, Elsevier, v. 84, n. 9, p. 1424–1435, 2011.

HUTCHINSON, J. E.; WHITTLE, J.; ROUNCEFIELD, M. Model-Driven Engineering Practices in Industry: Social, Organizational and Managerial Factors that Lead to Success or Failure. *Sci. Comput. Program.*, v. 89, p. 144–161, 2014.

ISO/IEC/IEEE. *ISO/IEC/IEEE 42010 Systems and Software Engineering — Architecture Description*. 1. ed. [S.l.], 2011.

JANSEN, A.; BOSCH, J. Software Architecture As A Set of Architectural Design Decisions. In: IEEE. *Software Architecture, 2005. WICSA 2005. 5th Working IEEE/IFIP Conference on*. [S.l.], 2005. p. 109–120.

KITCHENHAM, B. A.; PFLEEGER, S. L. Principles of Survey Research Part 4: Questionnaire Evaluation. *ACM SIGSOFT Software Engineering Notes*, ACM, v. 27, n. 3, p. 20–23, 2002.

KLEIN, J. How Does the Architect's Role Change As the Software Ages? In: IEEE. *5th Working IEEE/IFIP Conference on Software Architecture (WICSA'05)*. [S.l.], 2005. p. 141–141.

KLEIN, J. What Makes an Architect Successful? *IEEE Software*, IEEE, v. 33, n. 1, p. 20–22, 2016.

KOENIG, A. Patterns and Antipatterns. *The Patterns Handbook: Techniques, Strategies, and Applications*, v. 13, p. 383–383, 1998.

KRUCHTEN, P. Common Misconceptions About Software Architecture. *The Rational Edge*, v. 1, p. 1998–1998, 2001.

KRUCHTEN, P. A What Do Software Architects Really Do? *The Journal of Systems and Software*, ScienceDirect, n. 81, p. 2413–2416, 2008.

KRUCHTEN, P.; CAPILLA, R.; DUEÑAS, J. C. The Decision View's Role in Software Architecture Practice. *IEEE software*, IEEE, v. 26, n. 2, p. 36–42, 2009.

KRUCHTEN, P.; OBBINK, H.; STAFFORD, J. The Past, Present, and Future for Software Architecture. *IEEE software*, IEEE, v. 23, n. 2, p. 22–30, 2006.

LIKERT, R. A technique for the measurement of attitudes. *Archives of psychology*, 1932.

MICROSOFT. The Role of an Architect. *The Architecture Journal*, Microsoft, n. 15, 2008.

PFLEEGER, S.; KITCHENHAM, B. Principles of Survey Research Part 1"Turning Lemons into Lemonade. *Software Engineering Notes*, ACM SIGSOFT, n. 26, 2001.

RAHMAN, A. et al. Which Factors Influence Practitioners' Usage of Build Automation Tools? In: IEEE PRESS. *Proceedings of the 3rd International Workshop on Rapid Continuous Software Engineering*. [S.l.], 2017. p. 20–26.

RAUNAK, M. S.; BINKLEY, D. Agile and Other Trends in Software Engineering. In: IEEE. *2017 IEEE 28th Annual Software Technology Conference (STC)*. [S.l.], 2017. p. 1–7.

REHMAN, I. et al. Roles And Impacts of Hands-on Software Architects in Five Industrial Case Studies. In: ACM. *Proceedings of the 40th International Conference on Software Engineering*. [S.l.], 2018. p. 117–127.

SEI. *Community Software Architecture Definitions*. 2017. Disponível em: <http: //www.sei.cmu.edu/architecture/start/community.cfm>.

SEI. *Duties, Skills, and Knowledge of a Software Architect*. 2017. Disponível em: <https://www.sei.cmu.edu/architecture/research/previousresearch/duties.cfm>.

SHAW, M.; CLEMENTS, P. The Golden Age of Software Architecture. *IEEE software*, IEEE, v. 23, n. 2, p. 31–39, 2006.

SHERMAN, S.; HADAR, I. Toward Defining the Role of the Software Architect. In: *IEEE/ACM 8th International Workshop on Cooperative and Human Aspects of Software Engineering*. [S.l.]: IEEE, 2015. p. 71–76.

SOMMERVILLE, I. *Engenharia de Software*. São Paulo, Brasil: Pearson Education do Brasil Ltda, 2019.

SPINELLIS, D. The Changing Role of the Software Architect. *IEEE Software*, IEEE, v. 33, n. 6, p. 4–6, 2016.

TANG, A.; LAU, M. F. Software Architecture Review By Association. *Journal of Systems and Software*, Elsevier, v. 88, p. 87–101, 2014.

TANG, A. et al. Human Aspects in Software Architecture Decision Making. In: IEEE. *2017 IEEE International Conference on Software Architecture (ICSA)*. [S.l.], 2017. p. 107–116.

TOFAN, D. et al. Past and Future of Software Architectural Decisions – A Systematic Mapping Study. *Information and Software Technology*, Elsevier, n. 56, p. 850–872, 2014.

TYREE, J.; AKERMAN, A. Architecture Decisions: Demystifying Architecture. *IEEE software*, IEEE, v. 22, n. 2, p. 19–27, 2005.

VLIET, H. V. Software Architecture Knowledge Management. In: IEEE. *Software Engineering, 2008. ASWEC 2008. 19th Australian Conference on*. [S.l.], 2008. p. 24–31.

VLIET, H. van; TANG, A. Decision Making in Software Architecture. *Journal of Systems and Software*, Elsevier, p. 638–644, 2016.

WEINREICH, R.; BUCHGEHER, G. Towards Supporting the Software Architecture Life Cycle. *Journal of Systems and Software*, Elsevier, v. 85, n. 3, p. 546–561, 2012.

WEINREICH, R.; GROHER, I. The Architect's Role in Practice: From Decision Maker to Knowledge Manager? *IEEE Software*, IEEE, v. 33, n. 6, p. 63–69, 2016.

WOODS, E.; FAIRBANKS, G. The Pragmatic Architect Evolves. *IEEE Software*, IEEE, v. 35, n. 6, p. 12–15, 2018.

# Apêndices

# APÊNDICE A – E-mail *Survey* Trabalho do Arquiteto de Software

Olá, Bom dia! Encontrei seu perfil profissional por meio do LinkedIn! Sou Manoela dos Reis Oliveira, mestranda em Ciência da Computação da Universidade Federal de Sergipe e venho lhe convidar a participar de uma rápida pesquisa com o intuito de caracterizar o papel do arquiteto de software no Brasil.

Para participar, basta responder o questionário clicando nesse link: https://docs.google.com/forms/d/e/1FAIpQLSfiLJUfMEArg1g7Aah4Nsmoq8l30Vc397r50IRohoxKjSai6A/viewform.

Ele é constituído de perguntas sobre as atividades desempenhadas no seu cargo de arquiteto(a) de software. São apenas questões objetivas e prometo que não tomará muito o seu tempo.

Sua participação é muito importante e como forma de incentivar e demonstrar o seu valor, ao aceitar participar da pesquisa automaticamente participará de um sorteio de uma Smartband Original da Xiaomi, a Miband 2. O sorteio será realizado ao fim da etapa de coleta dos dados e o resultado será enviado para todos os e-mails dos participantes. As respostas poderão ser enviadas até o dia 22/08.

Peço ainda que caso conheça outros arquitetos de software repasse o questionário, dessa forma posso abranger um quantitativo ainda maior.

Desde já agradeço muito a sua participação e disponibilidade.

# APÊNDICE B – E-mail reduzido *Survey* Trabalho do Arquiteto de Software

Olá, sou mestranda em Ciência da Computação pela UFS e venho lhe convidar a participar de uma rápida pesquisa com o intuito de caracterizar o papel do arquiteto de software no Brasil e também a participar de um sorteio de uma smartband MiBand 2, basta clicar no link: https://goo.gl/X2fqFa. Obrigada!

# APÊNDICE C – E-mail Avaliação Proposta Plano de Ensino Arquitetura de Software

Olá, boa tarde!

Meu nome é Manoela e eu sou aluna de mestrado do Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Sergipe sob a orientação do Prof. Dr. Michel dos Santos Soares.

O tema central da minha pesquisa é o trabalho do Arquiteto de Software no Brasil e uma das etapas da minha pesquisa é a proposta de uma disciplina de arquitetura que englobe a teoria necessária para a boa prática do arquiteto de software.

Devido a sua experiência na área ficaria muita grata se o(a) senhor(a) pudesse avaliar a minha proposta.

Atenciosamente.

# APÊNDICE D – Versão inicial do plano de ensino de Arquitetura de Software

# PLANO DE CURSO

## IDENTIFICAÇÃO

**DISCIPLINA:** ARQUITETURA DE SOFTWARE **CÓDIGO:**

**C.H.:** 60 **Nº DE CRÉDITOS:** 4 (quatro) **P.E.L.:**

**PRÉ-REQUISITO(S):** Engenharia de Software

**TURMA:** **HORÁRIO:** **PERÍODO:**

**PROFESSOR:**

## EMENTA

Introdução à arquitetura de software – conceito, objetivos e escopo. Importância e impacto em um software. Processo para a construção de arquitetura de software. Elementos básicos de uma arquitetura de software. O papel do arquiteto de software no ciclo de desenvolvimento. Introdução a Padrões de Projeto. Estilos arquiteturais. Notações arquiteturais. Arquiteturas específicas de domínio e Frameworks. Modelagem. Linguagens de descrição de arquitetura (ADL). Métodos para a avaliação de arquiteturas de software. Ferramenta de suporte à construção, representação e avaliação de arquiteturas de software. Gerenciamento de riscos arquiteturais. Reutilização.

## OBJETIVOS

**1. Geral:**

- Oferecer aos alunos uma visão abrangente de arquitetura de software oferecendo métodos e técnicas para a representar e avaliar arquiteturas. Discutir o impacto de arquitetura de software dentro do ciclo de vida de desenvolvimento do software, bem como o papel do arquiteto de software no mesmo. Oferecer uma visão sobre o uso de ferramentas de apoio à construção, representação e avaliação de arquiteturas. Discutir as formas de mitigação de riscos e reutilização de arquiteturas.

**2. Específicos:**

- Aplicar conceitos de arquitetura de software para definir, documentar, comunicar e avaliar uma arquitetura de software.
- Representar a arquitetura de um sistema e seus componentes utilizando uma linguagem de descrição de arquitetura.
- Desenvolver em equipe uma arquitetura executável (versão inicial de um software), colocando em prática os conceitos de vistos em sala.

## CONTEÚDO PROGRAMADO (2H/AULA)

- AULA 01: Introdução. Plano de ensino da disciplina. Explicação sobre o projeto.
- AULA 02: Conceitos e objetivos da arquitetura de software. Importância e impacto em um software.
- AULA 03: O papel do Arquiteto de Software no ciclo de desenvolvimento de software.
- AULA 04: Processo para a construção de arquitetura de software. Elementos básicos de uma arquitetura de software.

- AULAS 05 e 06: Introdução a Padrões de Projeto. Estilos arquiteturais (pipe-and-filter, camadas, transações, publish-subscribe, baseado em eventos, cliente-servidor, MVC e outros).
- AULAS 07 e 08. Norma ISO/IEC/IEEE 42010 (concerns, stakeholders, visões, rationale, modelo arquitetural, etc).
- AULAS 09 e 10: Práticas.
- AULAS 11 e 12: Avaliação 1.
- AULAS 13 e 14: Notações arquiteturais (visões, representações, diagramas de componentes e outros). Exercícios
- AULAS 15 e 16: Arquiteturas específicas de domínio e Frameworks.
- AULAS 17 e 18: Modelagem: UML, MDA. Exercícios.
- AULAS 19 e 20: Linguagens de descrição de arquitetura (ADL). Ferramenta de suporte à construção, representação e avaliação de arquiteturas de software.
- AULA 21: Métodos para a avaliação de arquiteturas de software
- AULAS 22 e 23: Gerenciamento de riscos arquiteturais. Reutilização.
- AULAS 24 e 25: Práticas
- AULAS 26 e 27: Avaliação 2.
- AULA 28: Devolução de provas
- AULAS 29 e 30: Seminários do projeto.

## METODOLOGIA

A fim de propiciar uma aprendizagem efetiva por parte dos alunos, o conteúdo será trabalhado a partir de aulas expositivas, resolução de exercícios em sala de aula, de modo que a participação seja incentivada, e desenvolvimento de trabalhos práticos. Nas aulas práticas o grupo deve apresentar os determinados artefatos, de acordo com o planejamento inicial apresentado na 1.a aula da disciplina.

Como apoio extra sala de aula, os alunos poderão tirar suas dúvidas com o professor em horário definido.

## FORMA DE AVALIAÇÃO

- A1: 1 prova teórica, individual, sem consulta
- A2: 1 prova teórica, individual, sem consulta

A cada unidade, a nota será composta por 60% de nota de prova e 40% de nota do projeto. A nota do projeto será individual, dependendo da participação efetiva do aluno no projeto.

- Nota final = (A1+A2)/2
- O projeto da disciplina será feito em grupo mas será avaliado juntamente as avaliações individuais, de forma a incentivar que todos os alunos participem efetivamente do projeto.
- As datas de avaliação são divulgadas no 1º dia de aula, assim como demais atividades da disciplina e como a avaliação ocorrerá.

As aulas serão ministradas em sala de aula com auxílio de dispositivos áudio-visuais e quadro.

## BIBLIOGRAFIA

1. Básica:

- BASS, L.; CLEMENTS, P.; KAZMAN, R. Software Architecture In Practice. Addison-wesley, 2012.
- CLEMENTS, Paul et al. Documenting software architectures: views and beyond. Addison-Wesley, 2010.
- BUSCHMANN, Frank; HENNEY, Kevin; SCHMIDT, Douglas C. Pattern-oriented software architecture, on patterns and pattern languages. John wiley & sons, 2007.
- TAYLOR, Richard N.; MEDVIDOVIC, N.; ERIC M. Dashofy. Software Architecture: Foundations, Theory, and Practice. Wiley, 2009.
- ROZANSKI, Nick; WOODS, Eóin. Software systems architecture: working with stakeholders using viewpoints and perspectives. Addison-Wesley, 2011.

2. Complementar:

- LANZA, Michele; MARINESCU, Radu. Object-oriented metrics in practice: using software metrics to characterize, evaluate, and improve the design of object-oriented systems. Springer Science & Business Media, 2007.
- SILVEIRA, Paulo; SILVEIRA, Guilherme; LOPES, Sérgio; MOREIRA, Guilherme; STEPPAT, Nico; KUNG, Fabio. Introdução à Arquitetura e Design de Software – Uma Visão Sobre a Plataforma Java. Rio de Janeiro: Elsevier – Campus, 2012.
- CLEMENTS, P.; KAZMAN, R.; KLEIN, M. H. Evaluating Software Architectures: Methods and Case Studies, Addison-Wesley, 2001.
- BOOCH, Grady Booch. Handbook of Software Architecture. (disponível na web).

Cidade Universitária “Prof. José Aloísio de Campos”, XX de xxxxxx de 2019.

_____

**Professor**
Professor da Disciplina

# APÊNDICE E – Questionário do *Survey* sobre o Trabalho do Arquiteto de Software

4. **Qual o seu nível de escolaridade?** *
   *Marcar apenas uma oval.*

   - ◯ Superior Incompleto
   - ◯ Superior Completo
   - ◯ MBA/Especialização
   - ◯ Mestrado
   - ◯ Doutorado

5. **Número de anos de experiência em cargos relacionados a TI** *
   *Marcar apenas uma oval.*

   - ◯ 0-2
   - ◯ 3-5
   - ◯ 6-8
   - ◯ 9+

6. **Número de anos exercendo seu cargo atual?** *
   *Marcar apenas uma oval.*

   - ◯ 0-2
   - ◯ 3-5
   - ◯ 6-8
   - ◯ 9+

7. **Em quantos projetos você está envolvido simultaneamente no seu cargo atual?** *
   *Marcar apenas uma oval.*

   - ◯ 0-2
   - ◯ 3-5
   - ◯ 5+

8. **Em qual estado você trabalha atualmente?** *
   *Marcar apenas uma oval.*

- Acre
- Alagoas
- Amapá
- Amazonas
- Bahia
- Ceará
- Distrito Federal
- Espírito Santo
- Goiás
- Maranhão
- Mato Grosso
- Mato Grosso do Sul
- Minas Gerais
- Pará
- Paraíba
- Paraná
- Pernambuco
- Piauí
- Rio de Janeiro
- Rio Grande do Norte
- Rio Grande do Sul
- Rondônia
- Roraima
- Santa Catarina
- São Paulo
- Sergipe
- Tocantins

## Funções do arquiteto

De acordo com as atividades desempenhadas no seu emprego como arquiteto(a) de software, em que medida elas se encaixam dentro das atividades a seguir?

9. **Eu defino a arquitetura de software** *
   *Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

10. **Eu crio documentos que descrevem a arquitetura de software** *
*Marcar apenas uma oval.*

- ◯ Nunca
- ◯ Raramente
- ◯ Às vezes
- ◯ Muitas vezes
- ◯ Sempre

11. **Eu atuo como comunicador da arquitetura de software** *
*Marcar apenas uma oval.*

- ◯ Nunca
- ◯ Raramente
- ◯ Às vezes
- ◯ Muitas vezes
- ◯ Sempre

12. **Eu me certifico que todos estão utilizando a arquitetura proposta** *
*Marcar apenas uma oval.*

- ◯ Nunca
- ◯ Raramente
- ◯ Às vezes
- ◯ Muitas vezes
- ◯ Sempre

13. **Eu me certifico de que todos estão utilizando a arquitetura corretamente** *
*Marcar apenas uma oval.*

- ◯ Nunca
- ◯ Raramente
- ◯ Às vezes
- ◯ Muitas vezes
- ◯ Sempre

14. **Eu me certifico que a arquitetura seja definida em etapas para que haja progresso na organização antes que a arquitetura seja completamente definida** *
*Marcar apenas uma oval.*

- ◯ Nunca
- ◯ Raramente
- ◯ Às vezes
- ◯ Muitas vezes
- ◯ Sempre

15. **Eu me certifico que o software e a arquitetura estão em sincronia** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

16. **Eu me certifico que a administração entende os detalhes necessários da arquitetura** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

17. **Eu me certifico que a modelagem está sendo feita de forma correta, para garantir que atributos de qualidades serão atendidos** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

18. **Eu auxilio em questões como seleção de ambientes e ferramentas** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

## Funções do arquiteto

De acordo com as atividades desempenhadas no seu emprego como arquiteto(a) de software, em que medida elas se encaixam dentro das atividades a seguir?

19. **Eu identifico e interajo com os stakeholders para certificar que suas necessidades serão atendidas** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

20. **Eu convenço os stakeholders sobre o valor da minha solução arquitetural** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

21. **Eu me certifico de que a arquitetura não é apenas certa para operações, mas também para implantação e manutenção** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

22. **Eu resolvo disputas e analiso tradeoffs** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

23. **Eu resolvo problemas técnicos** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

24. **Eu mantenho e elevo a moral da equipe responsável pela arquitetura** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

25. **Eu compreendo e planejo a evolução da arquitetura** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

26. **Eu planejo a inserção de novas tecnologias** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

27. **Eu gerencio estratégias de identificação e mitigação de riscos associados à arquitetura** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

28. **Eu auxilio na compreensão do domínio do problema** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

## Funções do arquiteto

De acordo com as atividades desempenhadas no seu emprego como arquiteto(a) de software, em que medida elas se encaixam dentro das atividades a seguir?

29. **Eu aplico padrões já usados na resolução de problemas semelhantes** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

30. **Eu me mantenho atualizado(a) das últimas tendências em arquitetura de software** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

31. **Eu participo durante o planejamento dos projetos** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

32. **Eu gerencio indivíduos e equipes** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

33. **Eu estabeleço decisões arquiteturais** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

34. **Eu penso no impacto que as minhas decisões têm sobre a arquitetura atual** *
*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

35. **Eu atuo como consultor(a) da tecnologia** *

*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

36. **Eu estou envolvido em todas as etapas do ciclo de vida do software** *

*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

37. **Eu auxilio no marketing e futuras definições de produtos** *

*Marcar apenas uma oval.*

- Nunca
- Raramente
- Às vezes
- Muitas vezes
- Sempre

38. **Deseja enviar algum comentário ou observação?**

# TRABALHO DO ARQUITETO DE SOFTWARE

Olá, esta pesquisa está sendo realizada pela aluna do Mestrado em Ciência da Computação ofertado pelo Programa de Pós-graduação em Ciência da Computação da Universidade Federal de Sergipe, Manoela dos Reis Oliveira, sob orientação do Prof. Dr. Michel dos Santos Soares. O intuito desse questionário é auxiliar a caracterizar e documentar o papel desempenhado pelo arquiteto de software no Brasil.

Por essa razão, o(a) convido a participar da pesquisa, contribuindo com sua base de conhecimento prático na profissão. Sua participação consistirá em responder algumas perguntas de acordo com as funções desempenhadas no seu dia-a-dia como arquiteto(a) de software e/ou sistemas. Toda informação que o(a) Sr.(a) nos fornecer será utilizada somente para fins desta pesquisa e não será divulgada para terceiros.

A sua participação na pesquisa é voluntária. No entanto, como forma de valorizar e reconhecer a importância da sua participação, ao aceitar responder o questionário também estará automaticamente participando um um sorteio de uma Smartband Original da Xiaomi MiBand 2. O sorteio será realizado ao fim da coleta de respostas e o resultado divulgado para todos os e-mails participantes. Caso deseje desistir após ter iniciado não sofrerá qualquer prejuízo.

Peço ainda que caso conheça outros arquitetos possa repassar o questionário, seria de grande ajuda. Obrigada!

1. **Endereço de e-mail** *

2. **Deseja continuar?** *
   *Marcar apenas uma oval.*
   - ◯ Sim
   - ◯ Não *Pare de preencher este formulário.*

## Gostariamos de saber um pouco mais sobre você. Para isso selecionamentos algumas perguntas de caracterização.

3. **Qual o seu cargo atual?** *
   *Marque todas que se aplicam.*
   - ☐ Arquiteto de Software
   - ☐ Arquiteto de Sistemas
   - ☐ Analista de Sistemas
   - ☐ Gerencial
   - ☐ Líder Técnico
   - ☐ Desenvolvedor
   - ☐ Outro: