UNIVERSIDADE ESTADUAL DO NORTE DO PARANÁ

***CAMPUS* LUIZ MENEGHEL - CENTRO DE CIÊNCIAS TECNOLÓGICAS**

CIÊNCIA DA COMPUTAÇÃO

JOSÉ MARIA CLEMENTINO JUNIOR

ANÁLISE DA INFLUÊNCIA DA ENGENHARIA DIRIGIDA A MODELOS NA APRENDIZAGEM ORGANIZACIONAL EM EMPRESAS BRASILEIRAS DE DESENVOLVIMENTO DE SOFTWARE

Bandeirantes

2018

JOSÉ MARIA CLEMENTINO JUNIOR

ANÁLISE DA INFLUÊNCIA DA ENGENHARIA DIRIGIDA A MODELOS NA APRENDIZAGEM ORGANIZACIONAL EM EMPRESAS BRASILEIRAS DE DESESNVOLVIMENTO DE SOFTWARE

Trabalho de Conclusão de Curso submetido à Universidade Estadual do Norte do Paraná, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. André Luís Andrade Menolli

Bandeirantes

2018

JOSÉ MARIA CLEMENTINO JUNIOR

ANÁLISE DA INFLUÊNCIA DA ENGENHARIA DIRIGIDA A MODELOS NA APRENDIZAGEM ORGANIZACIONAL EM EMPRESAS BRASILEIRAS DE DESENVOLVIMENTO DE SOFTWARE

Trabalho de Conclusão de Curso submetido à Universidade Estadual do Norte do Paraná, como requisito parcial para obtenção do grau de Bacharel em Ciência da Computação.

COMISSÃO EXAMINADORA

_____
Prof. Dr. André Luís Andrade Menolli
UENP – *Campus* Luiz Meneghel

_____
Prof. Dr. Mauricio Massaru Arimoto
UENP – *Campus* Luiz Meneghel

_____
Profa. Dra. Daniela de Freitas Guilhermino Trindade
UENP- *Campus* Luiz Meneghel

Bandeirantes, 21 de novembro de 2018.

Dedico este trabalho aos meus pais Edilúcia e José Maria, sem os quais eu não teria tido forças de realizá-lo.

AGRADECIMENTOS

Agradeço primeiramente à Deus, por ter me sustentado em todos anos de minha vida e em especial nesta fase que está finalizando, sem Ele não teria chegado até aqui.

Agradeço aos meus pais Edilúcia e José Maria por me ofertar o privilégio de poder estudar durante estes quatro anos e principalmente por serem essenciais em todos os momentos durante a graduação.

Agradeço ao meu orientador André Menolli, que deu a oportunidade de participar de projetos desde o segundo ano da graduação, onde tive a oportunidade de aprender novos conhecimentos e conceito que não seriam aprendidos em sala de aula e por toda paciência nestes anos.

Agradeço aos meus amigos que me apoiaram durante esse período da graduação, em especial ao Diego, um irmão que o Projeto Rondon me ofereceu e sempre esteve presente, ao Super Fono vulgo Djalma pelo incentivo e amizade.

Aos meus amigos de classe: Sdayle, André, Felipe, Rafael e os demais que nesse período compartilhamos grandes momentos de alegria.

A todos os professores que participaram da minha formação acadêmica, em especial a banca, que sempre ajudou com o necessário para realização deste trabalho.

Enquanto houver vontade de lutar haverá esperança de vencer. (Santo Agostinho)

RESUMO

Com o passar dos anos, novas técnicas vêm sendo propostas para desenvolvimento de software com intuito de melhorar a qualidade no desenvolvimento. Uma das abordagens propostas é a engenharia dirigida à modelos (MDE) que prioriza a utilização de modelos nas etapas do desenvolvimento de software e visa principalmente diminuir a distância entre o domínio do problema e a implementação. Outro conceito importante é a aprendizagem organizacional, que prioriza a criação, transferência e armazenamento do conhecimento presente dentro das empresas. Sendo assim, este trabalho tem como intuito analisar e entender como o MDE pode influenciar a aprendizagem organizacional dentro de empresas de software. Para isto, foi realizada uma pesquisa, por meio de um questionário, aplicado a 29 empresas de desenvolvimento de software. Os resultados indicam que as empresas utilizam modelos principalmente nas etapas de desenvolvimento na qual existem poucas transformações entre os modelos. Sobre as empresas participantes, pode-se contatar que alguns modelos da MDE influenciam no processo de criação do conhecimento das organizações.

Palavras-chave: Engenharia dirigida à modelos. Aprendizagem organizacional. Processo do desenvolvimento de software.

ABSTRACT

In the course of the years, new approaches have been proposed in software development with its primary focus on improving the quality of the final product and increasing development productivity. The main approaches proposed, can highlight programming oriented aspects, which is an evolution of object-oriented programming, which aims primarily to improve the separation of interest, which directly affects the cohesion and coupling software. Another significant approach is driven development models, which has the purpose of automatically generating code through models. This approach facilitates the condition consistency between the model and the generated code, and streamlines the process of software development. Therefore, in order to use these approaches in a well-known domain, it proposes an approach and a tool for generating Hibernate persistence data codes based on model driven development and aspect-oriented programming. It is expected that the use of the tool, developers even being inexperienced with technology, can create data persistence productively and quality applications.

Keywords: Model-driven development. Aspect-oriented programming. Data persistence. Hibernate.

LISTA DE FIGURAS

LISTA DE QUADROS E TABELAS

LISTA DE SIGLAS

AO	*Organizational Learning*
BMMN	*Business System Planning*
BSP	*Business System Planning*
CIM	Computation-Independent Model
MDA	*Model-Driven Architeture*
MDD	*Model-Driven Development*
MDE	*Model-Driven Engineering*
MSDS	*Model-Driven Software Development*
NATO	*North Atlantic Treaty Organization*
OA	*Learning Organization*
OMG	*Object Management Group*
PIM	*Platform-independent Model*
PSM	*Plataform-specific Model*
UML	*Unified Modeling Language*

SUMÁRIO

1. INTRODUÇÃO

Neste capítulo introdutório é apresentada a contextualização e problematização do projeto, a justificativa, os objetivos e a organização do trabalho.

1.1 CONTEXTUALIZAÇÃO E PROBLEMATIZAÇÃO

No desenvolvimento de software, há uma grande quantidade de aplicações, sejam elas móveis, desktop ou web, as quais necessitam que suas etapas de desenvolvimento do software sejam bem estruturadas, desde a conversa inicial com o cliente, documentação, modelagem até a codificação em busca da satisfação do usuário final. No entanto, isto não é uma tarefa trivial, pois deve-se atentar ao máximo com as exigências de produtividade, qualidade, custos e prazos presentes no desenvolvimento de software, uma vez que os produtos e serviços de software estão se tornando cada vez mais complexos.

Uma razão importante que implica na dificuldade em desenvolver produtos de software complexos é a distância entre o domínio do problema e o domínio de implementação. O uso de processos, em especial aqueles que prezam pela utilização de modelos, procuram diminuir esta distância existente entre os domínios, presente em vários níveis de abstrações que são descritos nos modelos. Segundo France e Rump (2007) e Paludo (2016) o crescimento da complexidade dos produtos de software é o principal motivador para os esforços na industrialização do desenvolvimento de software.

Naur e Randell (1969) demonstraram que a distância entre o que se desejava do produto de software e o que realmente se entregava ocorria em diversas dimensões, até porque o usuário muitas vezes não sabe o que realmente a aplicação deve realizar ou também não consegue expressar os requisitos do sistema.

Apesar de toda dificuldade do passado ou até mesmo atualmente em alguns segmentos, a engenharia de software evolui constantemente com a formação de engenheiros de software sempre mais preparados para lidar com sistemas de software cada vez mais complexos.

Sommerville (2011) ainda contribui:

> [...]É claro que ainda há problemas tais como: software entregue com atrasos e com custos maiores do que os esperados. Mas, não pode-se deixar de citar os reais sucessos e os inovadores métodos e tecnologias de engenharia de software que foram desenvolvidos [...]

Uma das formas utilizadas para endereçar alguns dos problemas de especificação, documentação e modelagem no desenvolvimento de produtos de software é empregando a abordagem da engenharia dirigida a modelos, que fornece uma base para o desenvolvimento de produtos de software a partir de modelos que abrangem todo o ciclo de vida do desenvolvimento.

Brambila, Cabot e Wimmer (2012) definem o desenvolvimento dirigidos a modelos como uma abordagem que faz o uso de modelos como os artefatos primários do desenvolvimento de software e geralmente a implementação é gerada parcial ou completamente automática.

A arquitetura dirigida a modelos é uma visão particular do desenvolvimento dirigido a modelos, que é proposto pelo grupo OMG (*Object Management Group*) (OMG, 2003) que adota padronização na confecção dos modelos e das linguagens de transformação.

Já a engenharia dirigida a modelos é considerada com um nível de abstração maior do que o desenvolvimento dirigido a modelos, porque vai além das atividades de desenvolvimento e abrange outras tarefas do processo de engenharia de software (BRAMBILLA; CABOT; WIMMER, 2012). Resumidamente, a engenharia dirigida a modelos pode ser expressa conforme a seguinte equação: ” Modelos + Transformações = Software”.

Os modelos e as transformações são criados, manipulados em sua grande maioria por pessoas inseridas dentro da organização, que teoricamente conhecem o processo pois, etapas, padrões de desenvolvimento. No entanto, nem sempre isso acontece, como citado anteriormente, em alguns casos, as tarefas se concentram em apenas um único usuário, ou um usuário é responsável por vários artefatos dentro do desenvolvimento, podendo gerar sérios problemas tais como: a perda de conhecimento pela saída de um profissional qualificado e especializado e a falta de modelos (documentações) que são fundamentais para a compreensão do projeto pelos membros da equipe (MENOLLI, 2013).

Para minimizar esses problemas, é necessário mudar as práticas organizacionais, com intuito de aumentar a base de conhecimento das pessoas em relação ao processo de software e também o compartilhamento de informações da

base de dados para a organização em um âmbito geral. Segundo Menolli (2013), Nevis, Di Bella e Gould (1995), uma área que pode auxiliar quanto a esse objetivo é a aprendizagem organizacional, que é a capacidade ou os processos dentro da organização que tem o foco em manter ou melhorar o desempenho com base na experiência.

Para dar suporte ao conceito de aprendizagem organizacional, várias teorias e modelos de compartilhamento de conhecimento e aprendizagem foram criados, esses modelos descrevem os processos em níveis individuais ou organizacionais e servem de suporte para entender como se dá o processo de aprendizagem dentro da organização.

Somente a adoção individual da engenharia dirigida a modelos ou da aprendizagem organizacional, já apresenta benefícios para o processo de desenvolvimento de software (MAGALHÃES et al., 2011). Ao estimular a integração de ambas abordagens, pode-se ocorrer uma melhoria no processo de desenvolvimento (PALUDO, 2016).

1.2 OBJETIVOS

Este trabalho visa analisar quais modelos da engenharia dirigida a modelos são aplicados dentro das organizações desenvolvedoras de software, e como estes modelos influenciam o processo de aprendizagem organizacional. Além disso, pretende-se verificar a influência da engenharia dirigida a modelos sobre o processo de aprendizagem organizacional nas diferentes etapas de desenvolvimento de software em empresas brasileiras de desenvolvimento de software.

1.2.1 Objetivos Específicos

Como objetivos específicos podem ser destacados:

- Identificar e selecionar as principais técnicas que são utilizadas nas etapas de requisitos, projeto e construção de software em empresas de desenvolvimento de software;
- Identificar e selecionar as teorias da aprendizagem organizacional utilizadas nas empresas.

- Identificar e mapear como os conceitos do MDE podem influenciar na aprendizagem organizacional;
- Elaborar o instrumento de pesquisa;
- Selecionar as empresas participantes e aplicar o questionário;
- Realizar análise qualitativa e quantitativa dos dados obtidos por meio dos questionários.

1.3 JUSTIFICATIVA

Empresas desenvolvedoras de software cada vez mais necessitam de técnicas e ferramentas para buscar uma melhora na produtividade e qualidade em seus produtos de software. O cenário da fábrica de desenvolvimento vem melhorando com o passar dos anos, conforme os índices apontados por *Standish Group* (2010), uma organização que é especializada em pesquisa, avaliação de organizações voltada para área da tecnologia da informação.

É possível observar na Tabela 1.1.1 que existe uma evolução na indústria de software, principalmente se observar os projetos concluídos com sucesso. Mas ainda pode-se esperar um esforço considerável a ser empregado pelas organizações, para aumentar o índice de projetos com sucesso.

Tabela 1.1.1 Consolidação dos resultados do CHAOS REPORT – 1994 a 2010.

	1994	**1996**	**1998**	**2000**	**2002**	**2004**	**2006**	**2008**	**2010**
Sucesso	16%	27%	26%	28%	34%	29%	35%	32%	37%
Modificado	53%	33%	46%	49%	51%	53%	46%	44%	42%
Falha	31%	40%	28%	23%	15%	18%	19%	24%	21%

Fonte: Adaptado de (STANDISH GROUP, 2010).

Uma das formas de viabilizar que os projetos sejam concluídos com sucesso é definir e aplicar processos de desenvolvimento de software. Para tal, um dos principais fatores motivacionais desse trabalho é entender como empresas brasileiras de desenvolvimento de software promovem as práticas da utilização de modelos e de aprendizagem organizacional, buscando traçar um cenário da influência de ambos no desenvolvimento de software.

De acordo com o objetivo da engenharia dirigida a modelos, que é reduzir a distância entre os domínios do problema e da implementação, France e Rumpe (2007) colaboram:

> [...]engenharia dirigida a modelos (MDE) é primariamente preocupada com a redução da distância que há entre o domínio do problema e o domínio da implementação, através do uso de tecnologias que apoiam a transformação sistemática das abstrações do nível do problema em implementações de software [...] (FRANCE; RUMPE, 2007, p. 38).

Para diminuir esta distância, o MDE prioriza o uso de modelos que descrevem os sistemas completos, favorecendo melhores práticas de desenvolvimento utilizadas dentro das empresas, apresentando os históricos de implementações desenvolvidas anteriormente, os problemas encontrados e soluções tomadas.

Com isto, é necessário que as organizações estabeleçam padrões e especificações de cada projeto, documentando as etapas do ciclo de vida do software, as ações realizadas e as dificuldades que foram encontradas no andamento dos projetos, sendo assim a engenharia de software é um mediador para que toda esta documentação possa ser armazenada e reutilizada, com intuito de evitar os mesmos erros cometidos anteriormente. Neste âmbito a aprendizagem organizacional pode contribuir com a engenharia de software.

A aprendizagem organizacional contribui em vários fatores dentro da organização tais como: auxiliar o processo de treinamentos de novos membros das equipes, armazenar e gerenciar todo conhecimento gerado pela equipe nos projetos desenvolvidos dentro da organização, estabelecer padrões de utilização das ferramentas necessárias para o desenvolvimento do software.

1.4 ORGANIZAÇÃO DO TRABALHO

O Capítulo 1, aqui apresentado, visa contextualizar o leitor, sobre do que se trata o trabalho; justificar e exibir o objetivo geral e os específicos.

No Capítulo 2 é apresentada a fundamentação teórica que contém os seguintes conceitos: Engenharia Dirigida a Modelos; Etapas do Desenvolvimento de Software e Aprendizagem Organizacional.

No Capítulo 3 é apresentado um posicionamento metodológico, bem como é apresentada a estrutura detalhada da pesquisa, com suas propostas iniciais baseado na investigação bibliográfica.

No Capítulo 4 é apresentada a formulação do instrumento de pesquisa que foi utilizado no trabalho.

No capítulo 5 é apresentado os resultados obtidos na pesquisa.

No capítulo 6 é apresentada a discussão dos resultados com foco nos objetivos do trabalho.

2. REVISÃO BIBLIOGRÁFICA

No capítulo a seguir são descritos os conceitos de engenharia dirigida a modelos, aprendizagem organizacional e as etapas do desenvolvimento de software.

2.1 ENGENHARIA DIRIGIDA A MODELOS

O termo engenharia dirigida a modelos (MDE) usualmente é utilizado para descrever abordagens que utilizam ou geram modelos abstratos de aplicações, para que possam ser transformados em implementações concretas (PALUDO, 2016). Um dos objetivos do MDE é buscar resolver o obstáculo da crescente complexidade dos produtos de software, que são compostas por tecnologias que não são suficientes para resolvê-la.

Stahl e Vöelter (2006) apresentam uma discussão sobre as possíveis diferenças entre a abordagem “baseada” em modelos e a “dirigida” a modelos, afirmando que a abordagem dirigida a modelos concebe aos modelos um papel central e ativo, e ainda adicionam que os modelos são tão importantes quanto os códigos fontes da aplicação. Se faz necessário sensibilizar os envolvidos no desenvolvimento sobre a importância que se procura atribuir aos modelos nesta abordagem de desenvolvimento (PALUDO, 2016). Um modelo pode ser descrito como uma visão abstrata de uma aplicação, que deixa de lado alguns detalhes e que podem ser compostos por modelos complementares, para representar a estrutura, interações, contexto e comportamento da aplicação (SOMMERVILLE, 2011).

France e Rumpe (2007) também descrevem um modelo como uma abstração de alguns aspectos que são desenvolvidos para atender objetivos específicos, tais como: apresentar a compreensão de uma pessoa descrevendo os aspectos do sistema ou apresentar informações que podem ser exploradas por processamento digital.

De acordo com organização internacional que aprova padrões abertos para aplicações orientadas a objetos (OMG) a especificação da arquitetura dirigida a modelo tem a seguinte definição:

> [...] Um modelo de um sistema é a definição ou especificação daquele sistema e seu ambiente para um propósito específico. Um modelo é frequentemente apresentado como uma combinação de desenhos e texto. O texto pode estar materializado em uma linguagem de modelagem ou em linguagem natural. [...] (OMG, 2003, (p. 2-2, tradução nossa).

Os modelos possuem um papel mais representativo dentro do MDE, comparado com o desenvolvimento tradicional que são usados apenas como artefatos, com objetivo de documentar as decisões de um projeto, o que pode trazer problemas em projetos que possuem complexidade alta.

Por isso, France e Rumpe (2007) explicam que o MDE deve se preocupar primeiramente com a redução da distância entre o domínio do problema e entre o domínio da implementação, por meios dos modelos que transformam as abstrações do nível do problema em implementações de software.

Os modelos estão inseridos na parte central do desenvolvimento, e por meio dele, o código fonte, teste e documentações são derivados (RECH; BUNSE, 2009). Portanto, o modelo pode ser considerado uma abstração de uma realidade, com intuito de definir uma vertente de um domínio específico. Rech e Bunse (2009) afirmam que um modelo é composto por outro modelo, que recebe o nome de “metamodelo”, e que em aplicações mais complexas, são formados por vários modelos.

Existem variâncias quando se trata do uso das siglas que especificam as áreas da engenharia dirigida a modelos, como: MSDS (*Model-Driven Software Development*) que implica o desenvolvimento de software dirigida a modelo, que possui uma representação pouco precisa, o MDD (*Model-Driven Development*), que implica no desenvolvimento dirigido a modelos, o MDA (*Model-Driven Architecture*) que implica na arquitetura dirigida a modelos (STAHL; VÖELTER, 2006).

O MDD implica em desenvolver códigos a partir de modelos de alto nível, uma das suas funcionalidades é a transformação de um modelo de software em código executável ou até mesmo em outros modelos mais evoluídos, seja integralmente ou parcialmente (MELO, 2016). O MDD auxilia os programadores e os demais envolvidos no desenvolvimento de software, pois os códigos são gerados

automaticamente sem que haja a necessidade de serem produzidos totalmente de forma manual. Segundo Jiang, Zhang e Miyake (2007), o MDD é uma evolução da engenharia de software, no que contribui para resolver um grande problema no desenvolvimento de software a incompatibilidade entre o modelo e o código criado, além de contribuir com a padronização dos códigos.

Já o MDA de acordo com Belix (2006) é uma estrutura ou arquitetura para o desenvolvimento de sistemas. Ele tem a finalidade de separar as decisões orientadas ao negócio das decisões orientadas a plataforma e implementação, permitindo assim maior flexibilidade durante as fases de especificação e desenvolvimento do sistema. Uma das diferenças entre o MDD e o MDA, é que o MDA envolve todas as etapas de desenvolvimento.

Segundo a OMG (2003) a cada ano que passa fica evidente que o MDA foi um dos proponentes do crescimento das pesquisa e aplicações envolvendo o MDE. Mesmo que a nomenclatura na maioria das vezes não é padronizada, é possível observar uma maior concentração sobre a arquitetura dirigida a modelos, recaindo sobre o projeto e implementação.

Como mencionado anteriormente, o MDE está preocupado com a redução da complexidade entre o domínio do problema e o domínio da aplicação, visando criar modelos abstratos, para que possam ser transformados em implementações concretas.

Com isto, o MDE se faz presente em todos os aspectos das etapas do desenvolvimento de software. Por esse motivo, dentre as diversas nomenclaturas envolvendo desenvolvimento de software ou engenharia de software dirigida a modelos, neste trabalho será utilizado o (MDE). Os demais termos MDD, MDSD, MDA, serão mencionados apenas quando alguma particularidade muito específica for identificada como necessária a ser explicitada.

2.1.1 Transformação de Modelos

Um dos mais importantes conceitos dentro do MDE é a separação que existe entre o modelo independente de plataforma (PIM- *Platform-Independent Model*) e o modelo específico de plataforma, pelo fato dos conceitos de negócios serem mais estáveis do que as tecnologias (STAHL; VÖELTER, 2006). A Figura 2.1, apresenta o relacionamento entre as transformações dos modelos PIM e PSM.

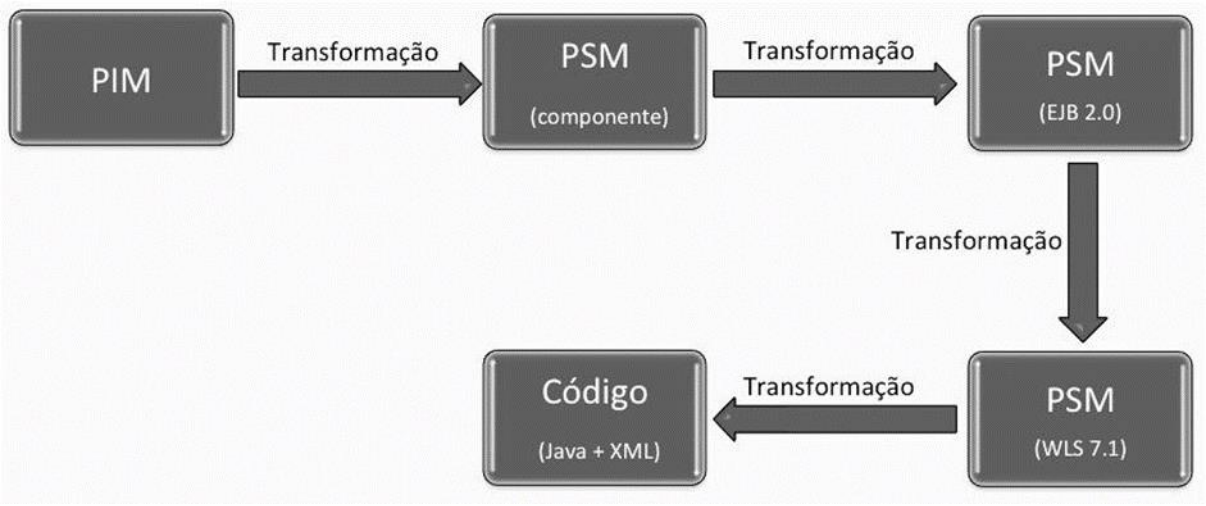

Figura 2.1. Sequência de transformação PIM e PSM, adaptado de (STAHL; VÖELTER, 2006)

A OMG (2003) define uma plataforma como um conjunto de tecnologias e subsistemas capaz de gerar um conjunto de funcionalidades para uma aplicação, por meio de padrões e interfaces, os quais podem ser utilizados por qualquer aplicação a suporte, desconsiderando os detalhes de implementação.

Os modelos independentes de plataforma retratam uma visão da aplicação, sem a necessidade de ser associada diretamente com as características da plataforma que hospeda a aplicação. Dessa forma, um modelo poder ser representado em vários tipos de plataformas semelhantes, abstraindo da tecnologia que será usada nos próximos passos do desenvolvimento (OMG, 2003).

Enquanto os modelos específicos de plataforma possuem todas as informações que são necessária para descrever o comportamento e a estrutura da aplicação, no modelo específico de plataforma (PSM - *Platform-specific model*) mesmo que não seja executável por si mesmo, deve ser um modelo informatizado e bem estruturado, para oferecer aos desenvolvedores a praticidade na utilização da implementação do código, ou gerar aplicações que irão conceber o código executável, buscando processos de transformação (BRAMBILLA; CABOT; WIMMER, 2012).

Modelo especifico de plataforma é definido também como uma visão de um sistema, do ponto de vista específico da plataforma. Ou seja, um PSM possui detalhes de como esse modelo usa a plataforma que irá hospedar a aplicação, e é obtido a partir da transformação de um modelo independente de plataforma. Quanto ao nível de detalhamento que o PSM contém, depende do objetivo que se deseja atingir (OMG, 2003).

Stahl e Völter (2006), consideram apenas os modelos independentes e dependente de plataforma, porém a OMG (2003) e (BRAMBILLA; CABOT; WIMMER, 2012) utilizam, também, o modelo independente da computação (CIM – *Computation Idenpendent Model)* que descreve o sistema de maneira independente da computação que será utilizada, mas não apresenta sua estrutura na aplicação. Este modelo está mais perto do especialista do domínio, buscando aproximá-lo, em conjunto com os seus requisitos de software, aos especialistas de projeto e construção dos produtos de software.

Com isso Brambilla, Cabot e Wimmer (2012) propõem um esquema de mapeamento entre os modelos, que traz informações tais como: onde os modelos podem ser encontrados no desenvolvimento de software e a sequência de suas transformações, como apresentado na Figura 2.2.

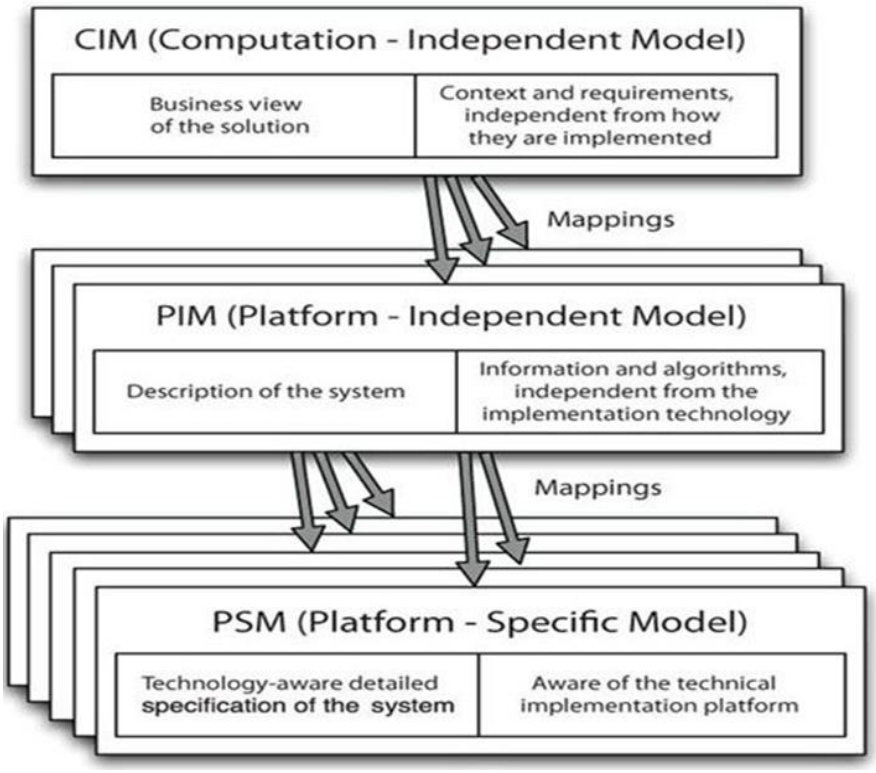

Figura 2.2 Mapeamento entre os modelos independentes de computação e de plataforma e os dependentes de plataforma, adaptado de (BRAMBILLA; CABOT; WIMMER, 2012)

Como apresentado na Figura 2.2, é visível que para existir uma transformação de um modelo para o outro é necessário criar uma transformação, que é classificada como um conjunto de regras de transformação, que em paralelo,

descreve como um modelo que usa a linguagem de origem, pode ser transformado em um modelo que usa a linguagem de destino (BENNETT; COOPER; DAI, 2010).

Para entender melhor, sobre o processo de transformação, se faz necessário identificar os artefatos que compõem cada modelo do MDA, para atingir este objetivo foi listado alguns artefatos que podem fazer parte da abordagem destes modelos, conforme é apresentado no Quadro 2.1.1.

Quadro 2.1.1 Artefatos presentes nos modelos do MDA (O autor)

Modelos do MDA	**Artefatos utilizados**	**Referências**
CIM	Diagrama de atividade (UML)	(DIAS; et al, 2006),(CALIARI (2007), (OLIVEIRA, 2008) (STEPHAN,2016)
	Português estruturado	(DIAS; et al, 2006)
	Casos de uso	(DIAS; et al, 2006) (BROWN; CONALLEN,TROPEANO 2005); (BELIX, 2006), (OLIVEIRA, 2008)
PIM	Tabela relacional Banco de dados	(BELIX, 2006)
	Diagrama de Sequência	(BELIX, 2006), (OLIVEIRA, 2008) (STEPHAN,2016)
	Digrama de Classe	(BELIX, 2006) (STEPHAN,2016)
	Diagrama de Fluxo	(HUSSAIN, 2017)
	Diagrama de Caso de Uso	(OLIVEIRA, 2008)
PSM	Geração de Código fonte	(BELIX, 2006)
	Geração de tabela relacional banco de dados	(BELIX, 2006)
	Diagrama Classe	(HUSSAIN, 2017)
	Geração de interface gráfica	(BROWN;CONALLEN,TROPEANO 2005) (Belix, 2006)

Além de apresentar como são realizadas as transformações entre os modelos CIM, PIM e PSM é necessário entender também em quais etapas do desenvolvimento de software estes modelos estão inseridos, uma breve discussão é apresentada a seguir.

2.2 ETAPAS DO DESENVOLVIMENTO DE SOFTWARE E O MDE

Com o passar dos anos juntamente com o avanço da tecnologia a complexidade dos produtos e serviço de software também cresceu, isto fez com que as organizações começassem a ter mais atenção no processo de desenvolvimento de software.

Sendo assim as organizações recorrem a engenharia de software, pois nela podem encontrar padrões arquiteturais, padrões comportamentais, padrões de implementações, ou seja, a engenharia de software, está preocupada com os aspectos da produção do software, desde a criação até a entrega e a manutenção. Para facilitar a compreensão dos aspectos da engenharia ela é subdivida em a áreas de conhecimento (SOMMERVILLE, 2007).

De acordo com Bourque et al (2004), são definidas quinze áreas de conhecimento, que são consideradas no ciclo de um software e nos processos da engenharia de software: requisitos de software, design de software, construção de software, gerenciamento de configuração de software, gerenciamento de engenharia de software, processo de engenharia de software, modelos e métodos de engenharia de software, qualidade de software, prática profissional de engenharia de software, economia de engenharia de software, fundações de computação, fundamentos matemáticos, fundações de engenharia.

Já para Sommerville (2007), o ciclo de vida do software é composto por 5 etapas e são realizadas de acordo com o projeto que será realizado, são elas: análise, projeto, implementação, testes e manutenção.

Bourque et al (2004), apresentam outra definição para as fases do ciclo de vida do software: requisitos de software, design de software e construção, que correspondem respectivamente com a definição de Sommerville (2007), análise, projeto e implementação.

Na sequência é apresentada uma breve descrição de cada área do ciclo de software que e também uma breve revisão bibliográfica sobre a relação entre etapas do ciclo de software e a engenharia dirigida a modelos.

Requisitos de software: na descrição mais simples o requisito de software é uma propriedade que deve ser exibida de certa forma para resolver um problema do mundo real (BOURQUE et al, 2004). É neste momento em que são firmados os acordos que entre os envolvidos no projeto quanto aos requisitos do sistema, também como todos os limites do projeto, estimativa de custo e tempo de desenvolvimento. Bourque et al (2004), divide a etapa de requisito de software em quatro partes: elicitação, análise, especificação e validação de requisitos.

A elicitação está relacionada com a origens dos requisitos de software e como que o engenheiro pode coletá-los. Sendo fundamentalmente uma atividade, a qual as partes interessadas são identificadas e também são estabelecidas relações entre a equipe de desenvolvimento e o cliente, também chamada de “captura de requisitos”.

Na análise de requisitos é onde são detectados e resolvidos os conflitos entre os requisitos, descobrir quais são os limites do software como ele será distribuído entre os membros da organização.

A especificação de requisitos de software estabelece a base para o acordo entre clientes e contratados ou fornecedores, sobre o que o produto de software deve ou não fazer. Os requisitos de software geralmente são escritos em linguagem natural, mas, na especificação podem ser complementados por descrições formais ou semiformais. A seleção de notações apropriadas permite que requisitos e aspectos particulares da arquitetura de software sejam descritos de forma mais precisa e concisa do que a linguagem natural, sempre buscando a maior precisão na descrição.

A validação dos requisitos de software usada para certificar se o engenheiro de software responsável pelo projeto compreendeu os requisitos do cliente. Também é importante para verificar se um documento de requisitos está de acordo com os padrões da empresa e dar sequência a próxima etapa do desenvolvimento: o design de software.

Design de Software: No design de software é desempenhado um papel importante dentro do desenvolvimento de software: durante o design de software, engenheiros de software produzem vários modelos, que são utilizados para auxiliar a

implementação da aplicação. Os modelos são analisados e avaliados com intuito de determinar se eles vão ou não cumprir os requisitos de software. Existem muitas notações para representar os artefatos de design de software. Alguns são usados para descrever a estrutura e organização de um projeto, outros para representar o comportamento do software.

Construção de Software: refere-se a criação detalhada do software através da combinação, decodificação, verificação, teste de unidade integração e depuração, a construção está ligada diretamente com o design de software porque é ele que dá o suporte das entradas necessárias para a construção.

Foi elaborada uma breve pesquisa literária afim de buscar relações entre a engenharia dirigida a modelos com as fases de desenvolvimento de software, conforme é apresentado no Quadro 2.2.2.

Quadro 2.2.2 Fases do Desenvolvimento de Software e o MDE (O autor)

Área do conhecimento	**Tópicos**	**MDE**	**Artigos**
Requisitos	Elicitação, Análise, Especificação e Validação	CIM	(BELIX,2006) (CHAVES; 2004) (FRANCE;RUMPE, 2007)
Design Software	Estrutura dinâmica, Estrutura estática	PIM, PSM	(NAZATTO, 2011) (BELIX,2006) (FRANCE;RUMPE, 2007)
Construção	Codificação, implementação	PSM	(BELIX,2006) (KLEPPE, Anneke G. et al, 2003)

A seguir são exemplificadas como se dá a relação entre os requisitos e o modelo CIM, os designs e o modelo PIM e a Construção e o PSM.

Requisitos e o CIM: como já dito anteriormente, a etapa de requisitos se

preocupa com as fases iniciais do projeto, com a interação com o usuário, a concepção do produto, com as especificações dos sistemas após a interação inicial com o cliente até o momento em que a validação dos requisitos é finalizada.

Segundo Belix (2006) o modelo CIM é útil para realizar a ponte entre cliente e desenvolvedores, pois é acessível a ambos. Chaves (2004) diz que o enfoque do CIM está ligado no ambiente e nos requisitos do sistema, não provê qualquer detalhamento quanto a estrutura e ao comportamento do sistema. France e Rumpe (2007) ainda afirmam que é no CIM que são definidos um vocabulário em comum para o uso em outros modelos, pela possibilidade de ser um modelo informal de fácil entendimento para as pessoas da organização. O CIM fornece os artefatos necessários para a construção do próximo modelo CIM e respectivamente os requisitos fornecem artefatos para a fase de design de software.

Design de Software e o PIM: na fase de design de software são construídos os modelos para a verificar se os requisitos foram atendidos. Nesta fase são criados artefatos, tanto para demonstrar o comportamento do software ou com sua estrutura. O mesmo acontece com o modelo PIM, onde são detalhadas as características estruturais e comportamentais do sistema que são independente da sua implementação em uma plataforma específica (BELIX, 2006). Nazatto (2011) ainda complementa que os modelos PIM especificam os elementos baseados em abstrações computacionais de forma genérica, como objetos, atributos, mensagens tipos de dados comuns, controle de fluxo e etc. France e Rumpe (2007) corrobora, afirmando que no modelo PIM não são considerados aspectos voltados à plataforma e à tecnologia da aplicação, e geralmente esses modelos são representados por diagramas da linguagem UML (*Unified Modeling Language*). Os autores afirmam também que um modelo PIM pode ser capaz de gerar vários modelos PSM.

Construção de Software e o PSM: a fase de construção está ligada à implementação do produto de software, utilizando-se das regras definidas na fase de design de software.

É na fase de construção que é realizado a implementação do código fonte, dos arquivos de configurações, da interface gráfica e todos os esquemas e buscas do banco de dados de uma aplicação. O mesmo acontece no PSM, a entrada do PSM vem da saída do PIM. Belix (2006) afirma que PSM é o produto final da

arquitetura, onde é realizado o mapeamento dos modelos independentes da plataforma para implementação em uma plataforma específica, gerando saídas do tipo programas em código fonte ou objeto, arquivos de configuração e esquemas de banco de dados.

No PSM que são gerados também todas as características finais de uma aplicação, ou seja, os modelos que anteriormente eram modelos abstratos passam a se tornar aplicações reais, detalhes como: a plataforma de desenvolvimento, tecnologias e ferramentas já estão definidas e são utilizadas para transformar os modelos em serviços ou produtos executáveis.

2.3 Aprendizagem Organizacional

Atualmente, as empresas prezam muito pelo fundamento do conhecimento, e este cada vez mais, é compartilhado por indivíduos, grupos e organizações. Desta forma, a competência de criar, adquirir, distribuir e implantar o conhecimento tem surgido como uma capacidade organizacional fundamental (TAKEISHI, 2002) (TEECE; PISANO; SHUEN, 1997). Para alcançar um nível de satisfação é necessário não explorar apenas o conhecimento atual, mas também os novos conhecimentos, com o intuito de criar novas estratégias competitivas.

Portanto, é necessário armazenar, criar organizar e reutilizar o conhecimento existente, com o intuito de evitar erros passados. Uma área que pode auxiliar a atingir este objetivo é a aprendizagem organizacional que é descrita como a capacidade, ou processos dentro da empresa, destinados a manter ou melhorar o desempenho baseado em experiência (NEVIS; DI BELLA; GOULD, 1995).

Com isso as organizações começaram a valorizar o conhecimento gerado dentro da organização e também o conhecimento de seus funcionários. Este conhecimento é aplicado de diversas formas na empresa, como rotina, práticas de produção e nos relacionamentos.

Desta forma, surge a necessidade de criar e implantar processos que concebem, armazenem, organizem, disseminem e apliquem o conhecimento produzido e utilizado na empresa de forma geral, explícita e acessível a toda comunidade da organização.

A aprendizagem organizacional pode ajudar nesta implantação. De acordo com Senge et al (1994) a aprendizagem organizacional pode ser definida como um teste

constante da experiência, que pode ser transformada em conhecimento acessível para toda a organização. Para Souza (2014) e Menolli (2013) aprendizagem organizacional são os processos voltados a estabilizar ou melhorar o desempenho da organização baseado na experiência, com o intuito de atingir resultados que indicam a realização da aprendizagem.

A aprendizagem organizacional pode identificar quais os tipos de estruturas organizacionais, política de gestão de pessoas, lideranças, culturas, valores e competências estão presente dentro da organização, que influenciam positivamente ou negativamente no processo de aprendizagem.

Dentro da Aprendizagem Organizacional, existem duas terminologias na literatura, que usadas de forma incorreta podem gerar uma possível controvérsia: Aprendizagem Organizacional (*Organizational Learning*) (AO) e Organizações de Aprendizagem (*Learning Organization*) (OA).

De acordo com Levine e Monarch (1998), se entende por AO quando a aprendizagem se destina entre indivíduos e grupos dentro da organização e quando se fala de OA, entende-se a aprendizagem pela organização como um sistema total, na qual existem recursos para aprendizagem sistêmica, sem contar as atividades próprias de alguns indivíduos que podem ir ou vir.

Apoiado nos trabalhos de Senge (1994), Levine e Monarch (1998) e Garvin (1993) e Menolli (2013), este trabalho considera a aprendizagem organizacional, como um processo de mudança apropriada que sofre influência pela experiência do passado, e que está centralizada no desenvolvimento ou nas alterações de rotinas, sustentada pela memória organizacional. O conhecimento que a organização obtém é construído primeiramente a partir de um único indivíduo, a partir de suas particularidades, crenças e habilidades em seguida é transferida para um grupo de pessoas, após esse momento com a integração do conhecimento de um grupo é transformado em um conhecimento acessível para a organização como um todo.

Todavia, para que este conhecimento possa ser sistemático e confiável é necessário que o conhecimento que foi gerado por um único indivíduo se espalhe. Para tal, existem modelos que objetivam maneiras de o conhecimento ser compartilhado ou que ocorra a aprendizagem, tanto em nível individual ou organizacional.

2.3.1 Teorias da Aprendizagem Organizacional

Para dar sustentação aos conceitos de aprendizagem organizacional, na literatura existem teorias e modelos de compartilhamento de conhecimento e aprendizagem. Alguma são destacadas por Bjørnson e Dingsøyr (2008): a teoria de criação do conhecimento de Nonaka e Takeuchi, teoria de comunidades de prática de Wenger, o modelo de aprendizagem experiencial de Kolb e a teoria de aprendizagem de duplo circuito de Argyris e Schön.

Kolb afirma que a aprendizagem organizacional é baseada em experiências, então nomeia como aprendizagem experiencial (KOLB, 1984). Sua teoria apresenta um modelo holístico do processo de aprendizagem, visando dar ênfase ao papel central que a experiência desempenha no processo de aprendizagem. Ele propõe um modelo de quatro fases do ciclo de aprendizagem: divergente, assimilador, convergente e acomodador.

Uma teoria diferente é proposta por Argyris e Schön (1996), para eles existem duas formas de aprendizagem: a de ciclo simples e a de ciclo duplo. A de ciclo simples atua na observação dos acontecimentos; a de ciclo duplo atua na observação das estratégias e analisa também os fatores que influenciam os efeitos da aprendizagem.

A teoria de comunidades de prática de Wenger (1998) é dita como as comunidades que geram suas próprias práticas de aprendizagem, tais como rotinas, símbolos, histórias, rituais. Ele também cita a diferenciação das práticas entre pessoas físicas, comunidade e organização. Quanto às pessoas físicas a aprendizagem ocorre no decorrer do exercício e contribuindo para uma comunidade. Na comunidade, a aprendizagem é modelar a prática. E nas organizações, a aprendizagem é sustentar comunidades de prática associadas.

Já para Takeuchi e Nonaka (2008), a teoria básica da criação de conhecimento, em sua estrutura possui duas dimensões, a epistemológica e a ontológica, como apresentado na Figura 2.3.

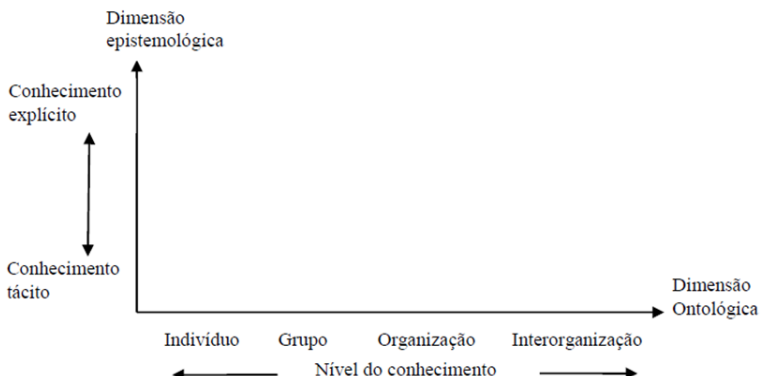

Figura 2.3 Duas Dimensões da Criação do Conhecimento. (TAKEUCHI; NONAKA, 2008)

Na perspectiva ontológica, o conhecimento é criado somente por indivíduos. Uma organização não consegue criar conhecimento sem os indivíduos, sendo assim a organização oferece suporte para indivíduos criativos proporcionando contexto para que seja possível desenvolver o conhecimento.

Na perspectiva epistemológica existe a diferenciação entre o conhecimento explícito e o conhecimento tácito. O conhecimento tácito é algo particular, exclusivo ao contexto, desta forma, há uma certa dificuldade para formalizar e comunicar. Já o conhecimento explícito ou também conhecido como “codificado”, diferentemente, se refere ao conhecimento que contém certa facilidade de ser transmitido em uma linguagem formal e sistemática (TAKEUCHI; NONAKA, 2008,).

Algumas das diferenças entre os dois tipos de conhecimentos são apresentadas no Quadro 2.3.

Quadro 2.3 Conhecimento tácito x Conhecimento explícito (TAKEUCHI; NONAKA, 2008)

Conhecimento tácito (subjetivo)	**Conhecimento explícito (objetivo)**
Conhecimento da experiência (corpo)	Conhecimento da racionalidade (mente)
Conhecimento simultâneo (aqui e agora)	Conhecimento sequencial (lá e então)
Conhecimento análogo (prática)	Conhecimento digital (teoria)

Porém, o conhecimento tácito e o conhecimento explícito não são absolutamente separados, mas entidades que se complementam. A criação do conhecimento é concebida através da interação social entre o conhecimento tácito e o explícito, o qual é denominado “conversão do conhecimento” (TAKEUCHI; NONAKA, 2008).

Consequentemente, Nonaka e Konno (1998) apresentaram o modelo de aprendizagem SECI, para que a conversão do conhecimento aconteça. O modelo SECI é um modelo espiral, o como resultado, o mesmo gera um conhecimento novo dentro da organização através da interação entre o conhecimento tácito e o explícito, conforme é apresentado na Figura 2.4.

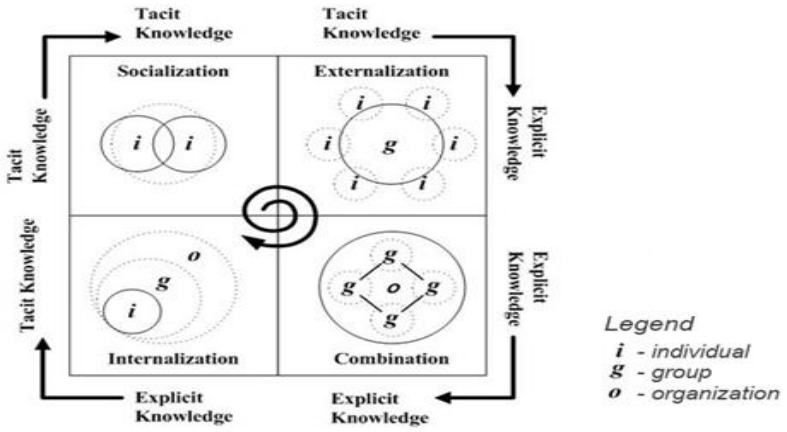

Figura 2.4 Espiral de Evolução do Conhecimento. Adaptado de (NOKANA e KONNO, 1998)

A respeito do processo de interação na criação do conhecimento, Nonaka e Konno (1998) atribuem quatro formas de conversão do conhecimento: socialização, externalização, combinação e internalização.

Socialização é o processo de conversão do conhecimento tácito de uma pessoa para o conhecimento tácito de uma outra pessoa. Geralmente é adquirida através do compartilhamento de experiência entre pessoas, por meio de observações de ações em uma relação do tipo “orientação”. No processo de socialização constrói-se o chamado “conhecimento compartilhado”.

Externalização é a conversão do conhecimento tácito para o conhecimento explícito. Para tornar o conhecimento tácito em conhecimento explícito é comum fazer o uso de metáforas, analogias, conceitos, modelos ou hipóteses, podendo utilizar a linguagem da fala, escrita e muitas vezes representações gráficas. O processo de externalização gera um tipo de conhecimento denominado “conhecimento conceitual”.

Combinação é o processo de conversão do conhecimento explícito criado por um indivíduo para o conhecimento explícito para a organização, combinação de diferentes conjuntos de conhecimento explícito. Ocorre quando os dados são manipulados por indivíduos (documentos, reuniões, e-mails, modelos) e, então, reconfigurados em forma de relatórios, ou outros tipos de documentos ou modelos.

Internalização é o contrário da externalização. É a conversão do conhecimento explícito para tácito, está diretamente ligado à aprendizagem pela prática, ou seja, aprender fazendo. As experiências através da socialização, externalização, combinação se tornam valiosas quando são internalizadas no conhecimento tácito do indivíduo. O processo de internalização gera um conhecimento chamado “conhecimento operacional”.

Apoiando nos trabalhos de Nonaka e Konno (1998) e Takeuchi e Nonaka (2008), este trabalho irá se sustentar na “ teoria de criação do conhecimento de

Nonaka e Takeuchi", pelo fato de um dos objetivos da pesquisa em si, é de descobrir como se dá o conhecimento dentro das empresas de software que utilizam as abordagens do MDE. Sendo assim, fica evidente a escolha pela mesma, que visa a aprendizagem do indivíduo como também da organização e principalmente que destaca o processo de criação do conhecimento e suas respectivas fases.

3. ESTRUTURA DA PESQUISA

O capítulo a seguir apresenta importantes conceitos a respeito da metodologia e métodos de pesquisa, com o objetivo de esclarecer ao leitor as opções metodológicas escolhidas e demonstrar as estratégias utilizadas para chegar ao objetivo do trabalho.

.

3.1 Caracterização da Pesquisa

De acordo com Barbosa (2017), é importante o desenvolvimento do modelo operacional e conceitual da pesquisa, dando a possibilidade de analisar os dados reais. Uma pesquisa não necessariamente deve seguir um único método, cada fase da pesquisa pode se aplicar diferentes métodos e ao final do trabalho obter um conjunto de métodos.

Desta forma, este trabalho pode ser classificado quanto ao seu objetivo como um estudo exploratório descritivo. Exploratório porque visa explorar quais tipos de técnicas do ciclo de software as empresas estão utilizando, mapeando também quais fases do MDA (CIM/PIM/PSM) se aplicam dentro das empresas. Descritivo devido ao fato de ser baseado em conhecimento teórico e em análises realizadas ao decorrer do trabalho, buscando descrever como essa proposta pode influenciar na aprendizagem organizacional (FREITAS, 2000).

Com isto, esta pesquisa é quantitativa, buscando verificar se o MDD está presente nas fases de desenvolvimento de software e verificar qual a influência que ele tem sobre a aprendizagem organizacional dentro de empresas brasileiras criadoras de software.

3.2 Estrutura da Pesquisa

Segundo Freitas (2000) pode-se considerar uma pesquisa como um questionário, quando se tem a capacidade de extrair informações de determinadas características por meio da análise de dados e ações de determinados grupos de pessoas, representado por um público alvo específico. Os questionários são

aplicados com o intuito de encontrar conclusões que possam ser generalizadas à população da qual a amostra foi retirada (MAFRA; TRAVASSOS, 2006).

O principal intuito do questionário neste trabalho é descobrir quais níveis de MDE são aplicados dentro das organizações e como estes níveis estão relacionados com a aprendizagem organizacional.

Sendo assim, para o cumprimento dos objetivos iniciais foi estabelecida uma estrutura de pesquisa (Figura 3.1). Basicamente, esta estrutura é composta por quatro etapas: Planejamento Inicial, Fase Exploratória, Desenvolvimento, Avaliação e Conclusão, as quais uma etapa pode ter conexão com as demais, demonstrada através das setas que especificam também os fluxos das atividades.

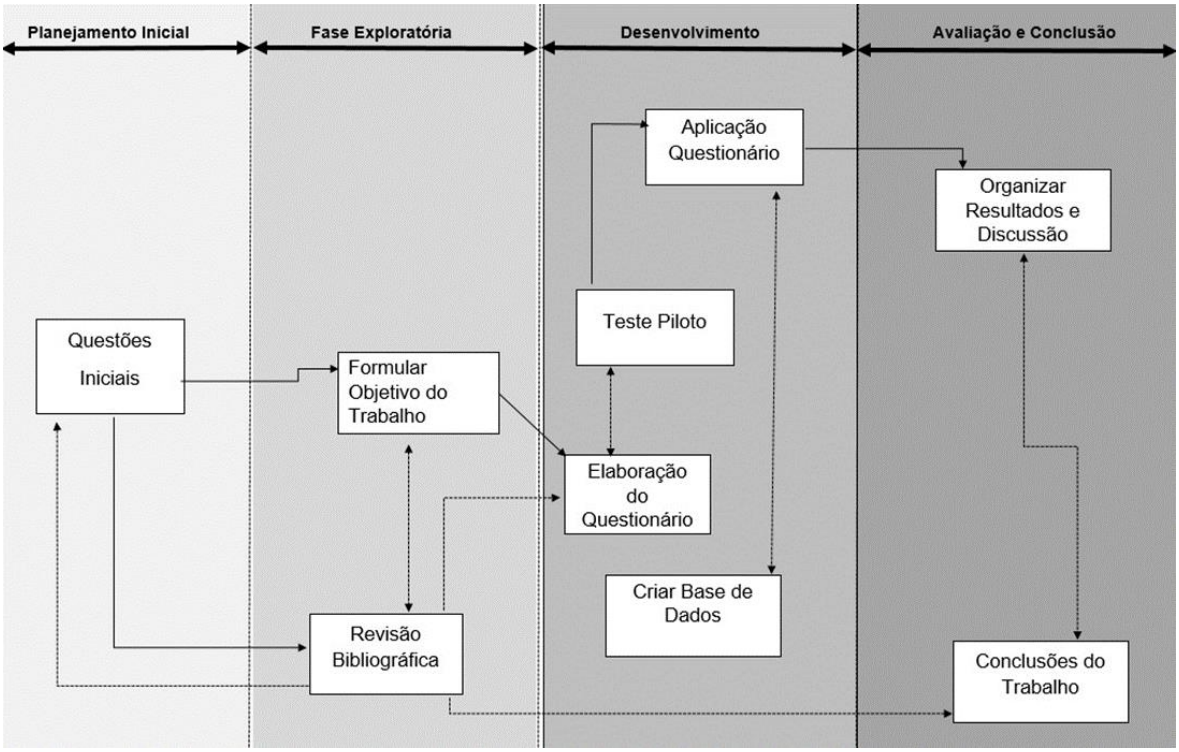

Figura 3.1 Estruturação da Pesquisa (O autor)

3.2.1 Planejamento Inicial

Logo após determinar a área de interesse, se fez necessário determinar questões iniciais para nortear o trabalho, tais como:

- Quais níveis do MDE são aplicados dentro do processo de criação de software da organização?
- Qual a influência dos modelos do MDE tem sobre a aprendizagem organizacional?

Entretanto, essas perguntas são apenas um direcionamento preliminar para o desenvolvimento do trabalho, visto que a área de estudo é bastante abrangente. Ao definir estas questões, delimitou-se o campo do estudo para atingir o objetivo a ser analisado. Desta forma, foi possível explorar o campo de estudo que será abordado na subseção a seguir.

3.2.2 Fase Exploratória

Esta etapa foi separada em duas partes distintas, visando um melhor entendimento do campo de estudo. A primeira, usando dados de fontes bibliográficas, e a segunda, a partir de dados coletados de pessoas, por meio da técnica de levantamento apoiada em questionários aplicados nas empresas.

Revisão Bibliográfica

É de grande importância realizar a revisão bibliográfica, pois o pesquisador tem a capacidade de poder identificar o conhecimento científico presente em uma área desejada, dando a possibilidade de planejar melhor o seu estudo, evitar o retrabalho de estudos já realizados e evitar erros já cometidos (MAFRA; TRAVASSOS, 2006).

Para descobrir quais são as etapas do desenvolvimento de software utilizam os modelos do MDA, foi realizado um estudo das técnicas de aprendizagem organizacional por meio de uma revisão bibliográfica, com o objetivo de apresentar os conceitos que são discutidos no trabalho (MENOLLI, 2012).

Definir Objetivo do Trabalho

As questões iniciais que foram definidas no escopo do trabalho, auxiliaram na identificação dos problemas que serão tratados. Além disso, o questionário, em paralelo com a revisão bibliográfica, proporciona uma visão geral do campo do estudo, fornecendo uma base para alcançar os objetivos do trabalho. Também foram definidos também o objetivo geral e os específicos do trabalho.

3.2.3 Desenvolvimento

Na fase de desenvolvimento são utilizadas todas as questões iniciais tratadas na fase de planejamento inicial. Além disso, todas as ideias apresentadas na fase exploratória são colocadas em práticas com intuito de responder às questões propostas.

Para alcançar o objetivo deste trabalho, foi necessário criar um processo de formulação e aplicação do instrumento de pesquisa que é apresentado na Seção 4.

3.2.4 Avaliação e Conclusão

A última fase da estrutura da pesquisa é a avaliação e conclusão, que para maior entendimento foram divididas em duas etapas: Organização e discussão dos resultados e Conclusões do Trabalho.

Organização e Discussão dos Resultados

Após a obtenção dos dados, deve-se organizá-los de forma legível para extrair informações que são utilizadas para auxiliar e atingir os objetivos estabelecidos. Após as informações disponíveis, foi realizada a discussão do trabalho.

Conclusão do Trabalho

Finalmente, após a coleta de dados, organização dos resultados e discussão, será possível concluir o trabalho, utilizando os resultados obtidos no questionário juntamente com todo o referencial teórico no decorrer da pesquisa, as conclusões finais do trabalho, poderão ser expressas, apresentando elementos que respondem aos objetivos iniciais do trabalho.

4. FORMULAÇÃO E APLICAÇÃO DO INSTRUMENTO DE PESQUISA

Nesta seção é discutida e apresentada toda a organização estrutural que envolve a elaboração do questionário. Para alcançar os objetivos do trabalho foram formuladas algumas questões baseadas em teorias e na revisão bibliográfica. Para compreender o processo de formulação e aplicação do instrumento de pesquisa na Figura 4.1 é apresentado o fluxo das atividades desenvolvidas no processo.

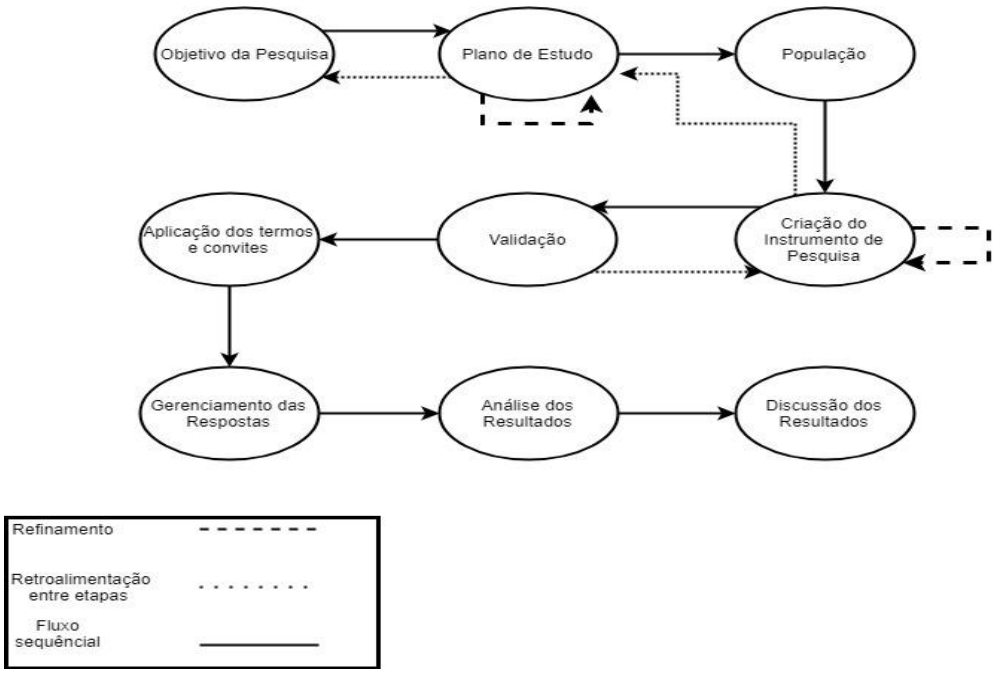

Figura 4.1 Estruturação do instrumento de pesquisa (O autor)

A seguir é apresentada cada etapa destacada na Figura 4.1 e o processo de transição entre elas.

4.1 Objetivo da Pesquisa

No cenário do desenvolvimento de software são elaboras e desenvolvidas muitas técnicas ou padrões que podem favorecer o desenvolvimento de software. Dentro deste contexto, uma área que pode ser explorada é a engenharia dirigida à modelos

que tem como principal objetivo a utilização de modelos que pode favorecer o desenvolvimento em todo o ciclo de vida do software.

Visto também que empresas desenvolvedoras de software produzem conhecimento a todo momento, desde conhecimento individual por parte de seus funcionários, conhecimento em grupo até o conhecimento como um todo da organização, se faz necessário entender como este conhecimento é desenvolvido e compartilhado pela organização. Com isso, o objetivo da pesquisa é identificar e analisar como a engenharia dirigida à modelos influencia a aprendizagem organizacional em empresas brasileiras desenvolvedoras de software. Para atingir o objetivo da pesquisa foram definidas três questões objetivas apresentadas no Quadro 4.1.

Quadro 4.1 Questões objetivas da pesquisa

Q1	Quais modelos e técnicas são utilizados pela empresa no processo de desenvolvimento de software?
Q2	Dentro do processo de desenvolvimento de software existe transformação entre modelos e como são mapeados?
Q3	Como os modelos utilizados pelas empresas podem influenciar no processo de criação da aprendizagem organizacional?

Para responder as questões objetivas apresentadas no (Quadro 4.1), foi elaborado um plano de estudo que é apresentado na próxima seção.

4.2 Plano de estudo

Para garantir que os objetivos da pesquisa fossem atingidos foi criado um plano de estudo que é representado pela Figura 4.2.

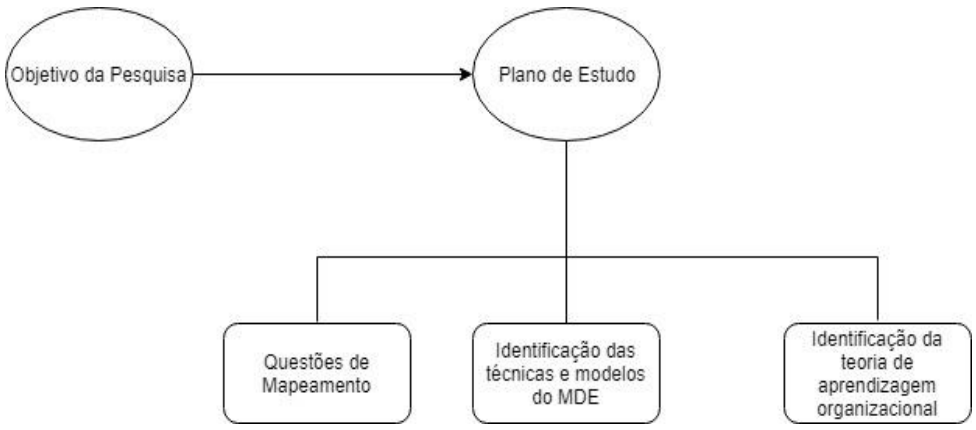

Figura 4.2: Estrutura do plano de estudo

A seguir serão apresentados como cada etapa (identificação das técnicas e modelos do MDE, identificação da teoria de aprendizagem organizacional e questões de mapeamento) do plano de estudo se relacionam em busca de atender ao objetivo da pesquisa.

4.2.1 Identificação das técnicas e modelos do MDE

Conforme apresentado nas Seções 2.1 e 2.2 do trabalho, foi realizada uma revisão bibliográfica com o com objetivo de identificar quais modelos do MDE são utilizados e aplicados dentro das etapas de desenvolvimento de software (requisitos, projeto e construção) e também como ocorre as transformações entre eles.

Se fez necessária a elaboração da revisão bibliográfica para a elaboração do mapeamento das perguntas que iriam compor o questionário e entender o cenário os quais os modelos são utilizados.

4.2.2 Identificação da teoria da aprendizagem organizacional

Conforme também apresentado na Seção 2.3 do trabalho, foi realizada uma revisão bibliográfica com o objetivo de estudar as principais teorias que compõem a aprendizagem organizacional.

Foi considerado principalmente o princípio de como o conhecimento é criado e compartilhado dentro das empresas, por isso o trabalho se apoia principalmente nas teorias de Nonaka e Konno (1998) e Takeuchi e Nonaka (2008). Também foram estudadas quais as relações entre os modelos do MDE dentro das etapas do desenvolvimento de software podem ser encontradas nas etapas da criação do conhecimento organizacional (socialização, externalização, combinação e internalização).

4.2.3 Questões de Mapeamento

Após a identificação das técnicas e modelos do MDE e a teoria da aprendizagem organizacional, foi desenvolvido o mapeamento das questões que responderiam as perguntas relacionadas ao objetivo da pesquisa. É importante destacar que o processo da criação das perguntas utilizadas no mapeamento passou por várias alterações.

As questões de mapeamento são utilizadas como base e direcionamento para criação das questões que foram elaboradas para a versão final do questionário,

portanto no decorrer do trabalho é apresentado o passo a passo e a relação entre: questões objetivo, mapeamento e questões do questionário final. No (Quadro 4.2) são apresentadas as relações entre as questões objetivo e as questões utilizadas no mapeamento:

Quadro 4.2 Questões Objetivo e Questões Mapeamento

Questões Objetivo	Questões Mapeamento
Q1: Quais modelos e técnicas são utilizados pela empresa no processo de desenvolvimento de software?	**Q1.1:** Quais modelos são mais comumente utilizados em cada fase (Requisito/Projeto)?
	Q1.2: Uma única pessoa é responsável por criar artefatos de etapas diferentes? Existe uma avaliação destes artefatos ou consulta para extrair soluções perante as dúvidas?
	Q1.3: Em qual fase ocorre o maior número de modificações?
	Q1.4: Qual a frequência que a empresa utiliza os modelos nas fases de desenvolvimento?
	Q1.5: A utilização de modelos pode gerar código finais automáticos ou semi-automaticamente?
	Q1.6 Quais modelos são considerados mais adequados em cada fase?
Q2: Dentro do processo de desenvolvimento de software existe transformação entre modelos e como são mapeados?	**Q2.1:** Como ocorre a transformação entre os modelos?
	Q2.2: Quais artefatos são responsáveis por gerar outros artefatos ocorre a transformação entre os modelos?
	Q2.3: Esses modelos são mapeados manualmente ou automaticamente nas fases de requisitos e projeto de software? Quais?
	Q2.4: A geração automática de código contribui no desenvolvimento?
Q3: Como os modelos utilizados pelas empresas podem influenciar no processo de criação da aprendizagem organizacional?	**Q3.1:** Como e quais os modelos favorecem o ensino e aprendizagem no processo de socialização?
	Q3.2: Como os modelos favorecem o ensino e aprendizagem no processo de combinação?
	Q3.3: Como os modelos favorecem o ensino e aprendizagem no processo de externalização?
	Q3.4: Como os modelos favorecem o ensino e aprendizagem no processo de internalização?

4.3 População

O Brasil conta com 11.237 empresas dedicadas ao desenvolvimento e à comercialização de software representando 1,7% do mercado mundial (ABES, 2018).

Especificamente 5.138 empresas são voltadas apenas para o desenvolvimento de software, distribuídas de acordo com o Quadro 4.3.

Quadro 4.3: Distribuição das empresas desenvolvedoras de software no Brasil (ABES, 2018).

Porte da empresa	Porcentagem:	Quantidade de funcionários
Micro	49,3%	Menos que 10 funcionários
Pequena	46,2%	De 10 a 99 funcionários
Média	4,0%	De 100 a 500 funcionários
Grande	0,5%	Mais que 500 funcionários

No Quadro 4.3 foram apresentados os portes e consecutivamente os valores das porcentagens que as mesmas representam e também a classificação de acordo com o número de funcionários. Para a aplicação do questionário foram consideradas empresas desenvolvedoras de software de pequeno, médio e grande porte, localizadas nos estados sul e sudeste, pois o sudeste representa 62,46 % e o sul 13,02 % de participação na distribuição do mercado de software e serviço do Brasil (ABES, 2018).

As buscas pelas empresas foram realizadas através de site de busca na internet. No total foram recolhidos os dados de 134 empresas que posteriormente foram convidadas a participar da pesquisa.

4.4 Criação do Instrumento de Pesquisa

O instrumento de pesquisa definido no trabalho foi o questionário, por possuir uma estratégia de aplicação pessoal (BOGDAN; BIKLEN, 1994) podendo ser encaminhado por correio, caixa de e-mail entre outros meios de comunicação. Outro fato é a facilidade na distribuição dos mesmos, tanto pela caixa de e-mail corporativo ou pessoal, dando a possibilidade também de entrar em contato diretamente com os funcionários e repassar o endereço em que o questionário estava hospedado.

A versão final do questionário foi composta por 31 questões, sendo elas 4 (quatro) questões abertas e 27 (vinte e sete) questões fechadas, envolvendo

aspectos da caracterização das empresas, modelos utilizados e aprendizagem organizacional.

Todas as questões foram elaboradas cuidadosamente com o intuito de não demostrar que as perguntas se tratavam especificamente sobre o MDE e aprendizagem organizacional. Para chegar à melhor versão do questionário algumas etapas foram necessárias ser seguidas, conforme é apresentado na Figura 4.3

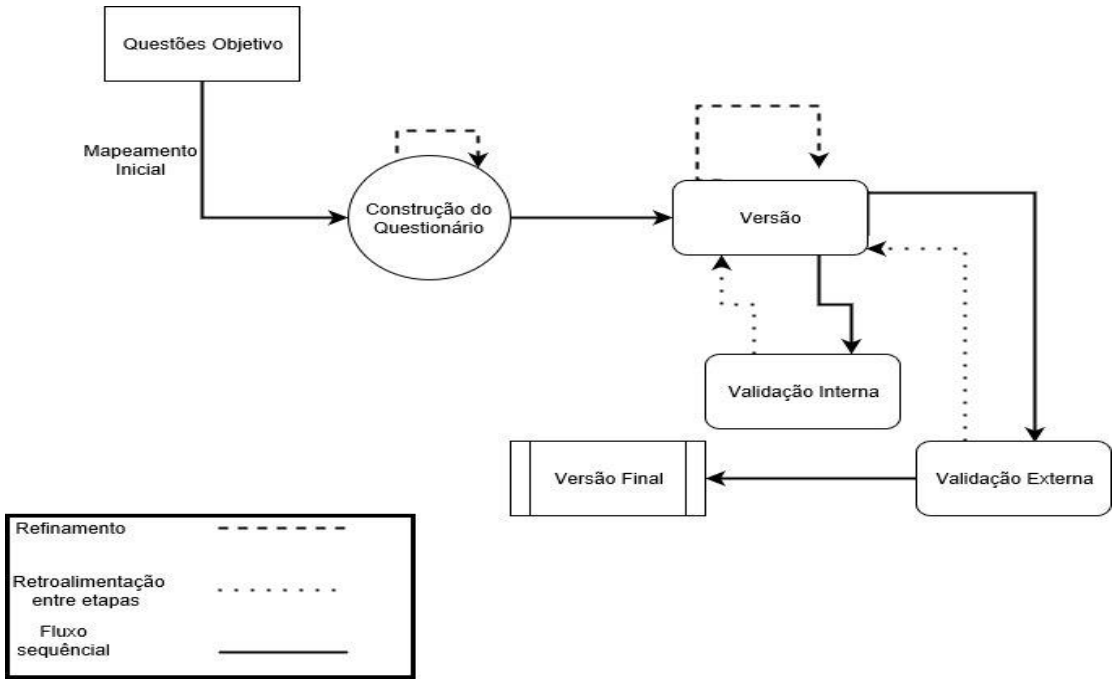

Figura 4.3: Processo de Criação do Instrumento de Pesquisa (O autor)

Conforme apresentado anteriormente as questões da versão final do questionário foram derivadas das questões do mapeamento que consequentemente responde as questões objetivo do trabalho. Para atingir a versão final do foram elaboradas 4 versões do questionário que serão descritas a seguir:

Questionário versão 1: A primeira versão do questionário tinha 39 questões sendo elas 5 abertas e 34 fechada. A plataforma utilizada para a aplicação do questionário foi o Google Forms. Sua validação ocorreu internamente dentro da instituição de ensino pelo professor orientador do trabalho, a qual necessitou passar por alterações.

Questionário versão 2: Após as alterações realizadas no questionário 1, a quantidade de questões permaneceu a mesma, foram realizadas alterações do tipo: melhora na escrita das questões e também nas alternativas e na estrutura do questionário. No final desta nova versão foi realizado um teste de validação

internamente com uma professora da área de engenharia de software, onde foi contabilizado o tempo de resposta do questionário que contabilizou 35 minutos.

Questionário versão 3: Após observado que o tempo de resposta do questionário era alto juntamente com algumas sugestões e críticas referente a estruturação do questionário ser muito cansativa, foi necessário melhorar sua estrutura. O questionário na versão 3 foi migrado para outra plataforma, o LimeSurvey.

LimeSurvey é uma ferramenta *open source, online* que permite a criação de perguntas e respostas dos diversos formatos, além de possuir uma gama de condições e validação de perguntas e respostas muito diversificada, também possui uma comunidade *online* para retirar as dúvidas e receber novas (SCHMITZ et Al, 2018).

Após desenvolvido o questionário no LimeSurvey o número de questões diminuiu de 35 para 29 questões devido a plataforma oferecer uma ótima opção de realizar combinações entre as questões. Ao término da versão 3 do questionário foi realizada uma validação interna por outro professor que atua na engenharia de software. Quanto ao tempo de resposta que anteriormente era de 25 minutos diminuiu pra 12 minutos com a nova estrutura criada no LimeSurvey. Após a validação, algumas considerações quanto as ortografias do texto foram alteradas.

Questionário versão 4: Após realizadas as 3 validações internas, o questionário passou por uma validação de um professor externo que também atua na área de engenharia de software, com ampla experiência na área de reuso de software e possui doutorado com ênfase em desenvolvimento dirigido à modelo, que sugeriu a adição de mais 2 questões e a alteração de algumas alternativas. Após esta validação o questionário chegou a sua versão final, com um total de 31 perguntas distribuídas da seguinte forma: 4 perguntas abertas e 27 perguntas fechadas. A estrutura do questionário foi dividida em três partes: caracterização da empresa, modelos e mapeamento.

Para ter um controle de gerência de resposta o mesmo questionário foi disponibilizado em dois links diferentes para os respondentes do e-mail e os dos Linkedin.

Para compreender como foram elaboradas as questões finais, surge a necessidade de apresentar os passos que foram realizados para atingir o objetivo:

1. primeiramente foram levantadas as questões objetivo do trabalho;

2. a seguir as questões de mapeamento para responder as questões objetivo;
3. e finalmente as questões finais que foram adicionadas ao questionário final a qual responde as questões do mapeamento e consequentemente as questões objetivo.

No Quadro 4.4 é apresentada a relação entre as questões do mapeamento com as questões finais do questionário.

Quadro 4.4: Relação entre as questões do mapeamento e questões finais do questionário(O autor)

(Parte 1- Continua)

Questões Mapeamento	**Questões do Questionário**
Q1.1: Quais modelos são mais comumente utilizados em cada fase (Requisito/Projeto)?	**Requisito: 2.1**-De que forma é realizada a interação com o cliente na fase de levantamento dos resquisitos de software? **Requisito: 2.3**- Após a conversa inicial com o cliente, em que tipo de documento as especificações dos requisitos de software são armazenadas? **Requisito: 2.4-** Qual tipo de notação a empresa utiliza na especificação dos requisitos? **Projeto: 2.5-** Dentre as representações existentes dos artefatos de projeto de software, quais são utilizadas para representar a visão estrutural (estática) na empresa? **Projeto: 2.6-** Dentre as representações existentes dos artefatos de projeto de software, quais são utilizadas para representar a visão comportamental (dinâmica) na empresa?
Q1.2: Uma única pessoa é responsável por criar artefatos de etapas diferentes? Existe uma avaliação destes artefatos ou consulta para extrair soluções perante as dúvidas?	**2.2-** O responsável por fazer o levantamento de requisitos de software é o mesmo encarregado por fazer a especificação dos requisitos de software? **2.10-** Em qual(is) situação(ões) normalmente os modelos são consultados pelos colaboradores? **2.11** Quando ocorrem mudanças nos projetos, quais artefatos são alterados para refletir as alterações necessárias?
Q1.4: Qual a frequência que a empresa utiliza os modelos nas fases de desenvolvimento?	**2.12-** Com que frequência e em quais etapas os projetos desenvolvidos na empresa são representados por modelos?
Q1.5: A utilização de modelos pode gerar código finais automáticos ou semi-automaticamente?	**2.13-** Na empresa, os códigos finais (códigos fonte, tabelas de banco de dados e interfaces gráficas) são gerados automaticamente ou semi-automaticamente a partir de quais artefatos?
Q2.1: Como ocorre a transformação entre os modelos?	**2.7-** Quando um artefato é utilizado para gerar outro artefato ou código, de que forma isto é realizado? **2.14-** Qual(is) ferramenta(s) a empresa utiliza para realizar a transformação de modelos em código fonte(automático ou semi-automático, total ou parcial?

Quadro 4.4: Relação entre as questões do mapeamento e questões finais do questionário(O autor)

(Continuação)

Questões Mapeamento	**Questões do Questionário**
Q2.2: Quais artefatos são responsáveis por gerar outros artefatos?	**2.8-** Na fase de projeto de software assinale a partir de quais artefatos (colunas) os artefatos estruturais (estáticos) (linhas) são criados? **2.9-** Na fase de projeto de software assinale a partir de quais artefatos (colunas) os artefatos comportamentais (dinâmicos) (linhas) são criados?
Q2.3: A utilização de modelos pode gerar código finais automáticos ou semi-automaticamente?	**2.3-** Após a conversa inicial com o cliente, em que tipo de documento as especificações dos requisitos de software são armazenadas? **2.7-** Quando um artefato é utilizado para gerar outro artefato ou código, de que forma isto é realizado? **2.14-** Qual(is) ferramenta(s) a empresa utiliza para realizar a transformação de modelos em código fonte(automático ou semi-automático, total ou parcial) **Requsito: 2.4-** Qual tipo de notação a empresa utiliza na especificação dos requisitos?
Q2.4: A geração automática de código contribui no desenvolvimento?	**3.5-** Avalie quanto a geração de código (automático ou semiautomático, total ou parcial) favorece os itens abaixo:
Q3.1: Como e quais os modelos favorecem o ensino e aprendizagem no processo de socialização?	**3.1-** Avalie as técnicas abaixo, considerando quanto cada uma favorece a interação do analista com o cliente para extração dos requisitos iniciais do sistema?

Quadro 4.4: Relação entre as questões do mapeamento e questões finais do questionário(O autor)

(Conclusão)

Questões Mapeamento	**Questões do Questionário**
Q3.2: Como os modelos favorecem o ensino e aprendizagem no processo de externalização?	**3.2-** Avalie as formas de documentação abaixo, como cada uma favorece o entendimento e descrição dos requisitos? (Coluna direita). **3.3-** Dentre as representações da visão estrutural (estática), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento? (Coluna direita). **3.4-** Dentre as representações visão comportamental (dinâmica), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento?(Coluna direita).
Q3.3: Como os modelos favorecem o ensino e aprendizagem no processo de combinação?	**3.2-** Avalie as formas de documentação abaixo, como cada uma favorece o entendimento e descrição dos requisitos? (Coluna direita). **3.3-** Dentre as representações da visão estrutural (estática), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento? (Coluna direita). **3.4-** Dentre as representações visão comportamental (dinâmica), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento?(Coluna direita).
Q3.4: Como os modelos favorecem o ensino e aprendizagem no processo de internalização?	**3.2-** Avalie as formas de documentação abaixo, como cada uma favorece o entendimento e descrição dos requisitos? (Coluna esquerda). **3.3-** Dentre as representações da visão estrutural (estática), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento? (Coluna esquerda). **3.4-** Dentre as representações visão comportamental (dinâmica), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento?(Coluna esquerda).

4.5 Validação

Conforme citado na Seção 4.4 o questionário passou por alterações durante o desenvolvimento da pesquisa, ao todo foram desenvolvidas 4 versões, que foram revisadas e validadas por quatro profissionais da área da engenharia de software, com os seguintes perfis:

Revisor 1: Validação Interna: O primeiro revisor foi escolhido pois acompanhou a pesquisa desde o princípio, possui doutorado em informática, professor em universidade pública tendo expertise em engenharia de software, gestão do conhecimento e aprendizagem organizacional, que foram temas tratados direta e indiretamente no trabalho.

Revisor 2: Validação Interna: O segundo revisor foi escolhido pois esteve presente durante o período de desenvolvimento da pesquisa, possui doutorado em ciência da computação, professor em universidade pública e possui expertise em engenharia de software, gerência de projetos e interação humano computador.

Revisor 3: Validação Interna O terceiro revisor foi escolhido pois também esteve presente durante o período de desenvolvimento da pesquisa, possui doutorado em ciência da computação e matemática computacional, professor em universidade pública e possui expertise em engenharia de software com ênfase em processo de software, qualidade de software.

Revisor 4: Validação Externa O quarto revisor foi escolhido pois possui doutorado em informática e sua tese de doutorado faz referência a abordagem MDE, além de professor ele é sócio de uma empresa de software e é membro da Comissão de Qualidade de Produto de Software da ABNT desde 1998, participando de graduação e elaboração de normas nacionais e internacionais.

4.6 Aplicação dos Termos e Convites

O trabalho de pesquisa possui como população alvo empresas que trabalham com o desenvolvimento de software. No momento inicial da pesquisa foram coletados 134 contatos de empresas que tinham informações do tipo: nome, cidade, telefone, endereço de e-mail e site da empresa.

Após coletado o nome das empresas representado pelo balão “Empresas de Desenvolvimento” na Figura 4.4, foi realizada uma busca na rede social “LinkedIn” em busca dos profissionais que trabalhavam nas respectivas empresas, de acordo com as características do respondente da pesquisa e em seguida foi encaminhada a solicitação de amizade para o mesmo, conforme apresentado no esquema da Figura 4.4.

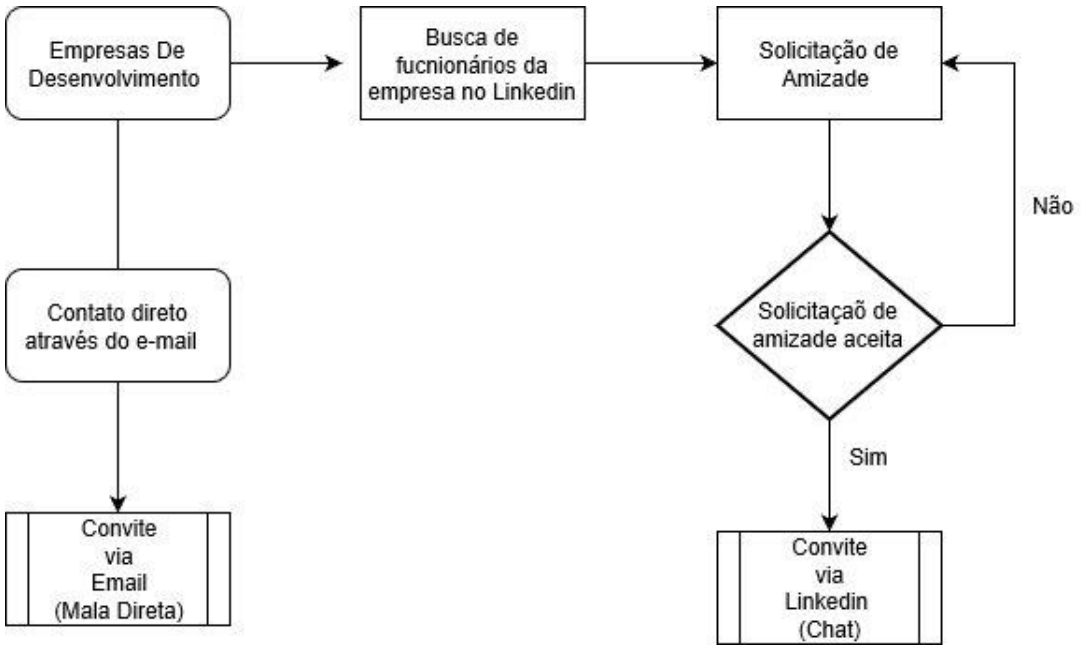

Figura 4.4 Aplicação dos termos e convites (O autor)

Foram coletados dados de 134 empresas desenvolvedoras de software para realizar o convite para a participação da pesquisa. Conforme apresentado na Figura 4-4, é possível observar que o convite para participação da pesquisa aconteceu de duas formas diferentes: e-mail corporativo e *LinkedIn*.

E-mail Corporativo: Após os dados coletados das empresas, foram criadas 3 planilhas no *Google Sheets* para realizar a organização da mala direta. Para auxiliar no processo de envio dos e-mail e realização da mala direta foi utilizado o YAMM (*Yet Another Mail Merge*), um complemento do Google que é uma ferramenta descrita como uma solução simples para enviar e-mails personalizados, boletins informativos ou formulários para vários destinatários.

A divisão das empresas em 3 planilhas no *Google Sheets* se deu pela a restrição da versão grátis do YAMM, por ele suportar apenas 50 envios diários de e-

mail. A divisão ficou da seguinte forma: Planilha 1 contendo 45 empresas, Planilha 2 com 44 e a Planilha 3 com 45 empresas somando ao total 134 empresas.

O Convite encaminhado para as empresas encontra-se no Apêndice B.

O questionário foi enviado duas vezes para as 134 empresas nos seguintes dias que são apresentados na Tabela 4.5:

Tabela 4.5 : Informações da data de envio do questionário por e-mail (O autor)

Identificação da planilha	**Número do envio**	**Data de envio**
Planilha 1	1	15/10/2018
Planilha 2	1	16/10/2018
Planilha 3	1	17/102018
Planilha 1	2	22/10/2018
Planilha 2	2	23/10/2018
Planilha 3	2	24/102018

O período de diferença entre os envios dos questionários foi de 7 dias, o motivo do reenvio se deu pela baixa taxa de resposta.

Linkedin: Inicialmente foi realizada uma busca do perfil das 134 empresas no Linkedin com intuito de descobrir o perfil de seus funcionários que teriam um potencial de responder o questionário de acordo com a função exercida dentro da empresa. Após descobertos os perfis, se fez necessário enviar uma solicitação de amizade, e em paralelo armazenou-se o link do perfil adicionado em uma base de dados para que consequentemente fosse encaminhado uma mensagem no chat da rede social.

Entre o período do envio da solicitação da amizade e envio do convite via chat teve um intervalo de 5 meses, para aguardar a solicitação de amizade ser aceita pelos usuários adicionados. Após este período e a análise de quantos usuários aceitaram o convite, obteve-se 200 perfis que aceitaram a solicitação de amizade.

O envio para os usuários aconteceu por meio do bate-papo e também aconteceram dois envios. O primeiro no dia 15 de outubro de 2018 e o segundo convite dia 28 de outubro de 2018. É importante lembrar que o segundo convite foi apenas enviado para aqueles usuários que apenas visualizaram a mensagem e não responderam.

O Convite encaminhado para os usuários do Linkedin encontra-se no Apêndice C.

4.7 Gerenciamento de Resposta

Como já citado anteriormente o questionário foi distribuído em dois ambientes diferentes e para cada ambiente foi atribuído um *link* do questionário distinto, justamente para ter o controle e o gerenciamento das respostas.

Controle via e-mail: para o convite realizado pela mala direta por e-mail com o auxílio da ferramenta YAMM foi possível observar algumas variáveis como: a quantidade de e-mails enviados; quantos e-mails foram abertos, entre outros. As informações de cada planilha da Tabela 4.6, seguindo a descrição de cada coluna são apresentadas a seguir assim como os dados obtidos dos questionários enviados por e-mail :

- *Planilha:* identificação das 3 planilhas que apresentavam o e-mail, nome e outras informações das empresas.
- *Enviado*: refere-se à quantidade de e-mail enviados.
- *Data envio:* refere-se a data que os e-mails foram despachados.
- *Aberto:* refere-se à quantidade de e-mail que foram abertos pelas empresas.
- *Clicado:* o conteúdo enviado no e-mail possuía o link do questionário online, esta variável refere-se à quantidade de empresas que abriram o link do questionário.
- *Respondido:* refere-se à quantidade de respostas ao e-mail enviado não propriamente o questionário.
- *Falhou:* a quantidade de e-mails que falharam no envio.
- *Descartado*: a quantidade de e-mails ao recebidos foram excluídos após recebidos.

Tabela 4.6: Gerenciamento dos e-mails de convite enviados (O autor)

Planilha	Enviado	Data envio	Aberto	Clicado	Respondido	Falhou	Descartado
Planilha 1	45	15/10/2018	15	2	0	2	0
Planilha 2	42	16/10/2018	18	6	0	2	0
Planilha 3	45	17/10/2018	21	3	2	3	0
Planilha 1	43	22/10/2018	16	2	0	2	0
Planilha 2	42	23/10/2018	15	3	0	2	0
Planilha 3	45	24/10/2018	13	2	0	3	0

Controle via Linkedin: O controle realizado por meio do Linkedin foi apenas a contagem dos convites enviados pelo chat. Foram enviados um total de 200 convites para os usuários da rede social.

Após finalizar a coleta dos questionários, a Tabela 4.7 apresenta a taxa de resposta de ambos os questionários disponibilizados entre os dias 15/10/2018 até 31/10/2018. As colunas apresentadas na tabela 4.7 são representadas por:

- *Id. Questionário:* a identificação do questionário o mecanismo de envio;
- *Incompleto*: refere-se à quantidade que o e-mail foi acessado, independente se ele foi ou não iniciado pelo respondente e que não foram completados.
- *Completo:* refere-se aos questionários que foram respondidos completamente.
- *Total:* refere-se a soma do número de questionários incompletos e incompletos.

Tabela 4.7: Quantidade de Respostas dos Questionários

Id. Questionário	Incompleto (%)	Completo (%)	Total (%)
Questionário E-mail	26 (93%)	2 (7%)	28 (100%)
Questionário Linkedin	309 (92%)	27(8%)	336(100%)

Os questionários do e-mail foram encaminhados para 134 empresas duas vezes o mesmo aconteceu para os 200 usuários do LinkedIn.

Após apresentada toda estruturação de pesquisa do trabalho e a formulação e aplicação do instrumento de pesquisa é apresentado os resultados obtidos com os questionários aplicados nas empresas desenvolvedoras de software.

5. RESULTADOS

Para melhor visualização dos resultados foram criadas algumas siglas para a representação dos dados nos gráficos e tabelas:

- **JAD:** *Joint Application Design*;
- **VORD:** Orientada a ponto de vista;
- **BSP:** *Business System Planning;*
- **BPMN:** *Business Process Modeling Notation*;
- **NUR:** A empresa não atua na fase de levantamento de requisitos de software;
- **NTD*:*** Não utiliza nenhum tipo de documento;
- **DA:** Diagrama de atividade;
- **KAOS:** Notação Kaos;
- **MCU:** Modelo de Caso de Uso;
- **HU:** História de usuário;
- **MF:** Método Formal;
- **DME:** Diagrama de Máquina e Estado;
- **NEDG:** Não especifica os documentos gerados;
- **ADL:** Linguagens de descrição de arquitetura;
- **DCL:** Diagrama de Classe.
- **DI:** Diagrama de Implantação
- **ER:** Diagramas Entidade Relacionamento
- **GE:** Gráfico de Estrutura
- **NUE:** Não é utilizado pela empresa.
- **IDLs:** Linguagens de descrição de interface;
- **PDLs:** Pseudocódigo e linguagens de design de programa;
- **DC**: Diagrama de comunicação;
- **DS**: Diagrama de sequência;
- **LEF**: Linguagem de representação formal;
- **NGA**: Não é gerado o artefato na empresa;
- **PRO**: Protótipo de tela;
- **SCRIPT:** Script de automatização;

- **EMF:** Eclipse Modeling Framework;
- **NPO:** Não posso opinar.

Os resultados serão apresentados de acordo com a organização do questionário nas seguintes seções: caracterização da empresa, modelos e mapeamento.

5.1 Caracterização da Empresa

Quanto à distribuição geográfica das empresas participantes em relação ao estado de localização constatou-se que 69% delas pertencem ao estado do Paraná, 28% São Paulo e 3 % ao Rio de Janeiro (Figura 5.1).

No que se refere ao tipo de capital das empresas participantes, o capital privado representa 86%, enquanto o capital misto 10% e o capital público 3% das empresas respondentes (Figura 5.2).

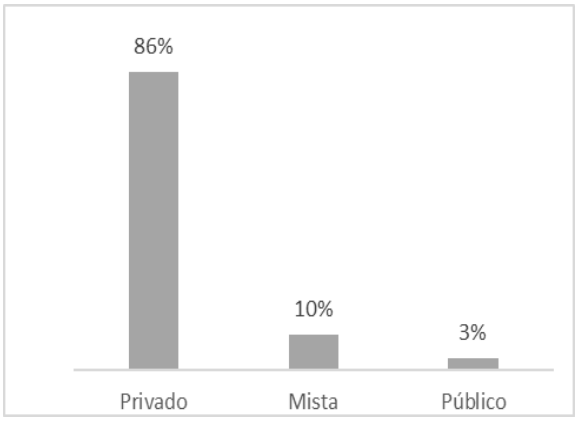

Figura 5.2: Tipo de capital da empresa

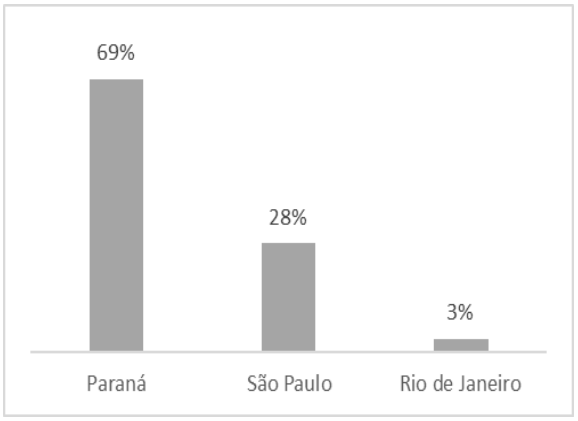

Figura 5.1: Distribuição geográfica

Também foi levado em consideração o perfil das empresas de caráter nacional ou internacional, constatou-se 69% delas são nacionais e 31% internacionais (Figura 5.3). Quanto à distribuição dos clientes que as empresas atendem, 17% representam apenas clientes nacionais, América do Sul 27%, América do Norte e Europa 16% ambas (Figura 5.4).

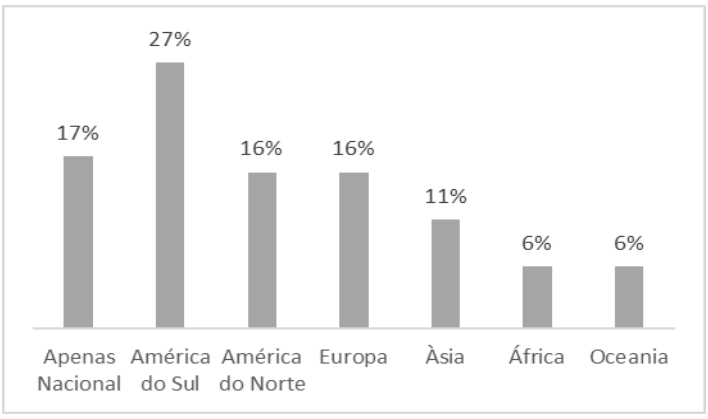

Figura 5.4: Distribuição dos clientes atendidos pela empresa

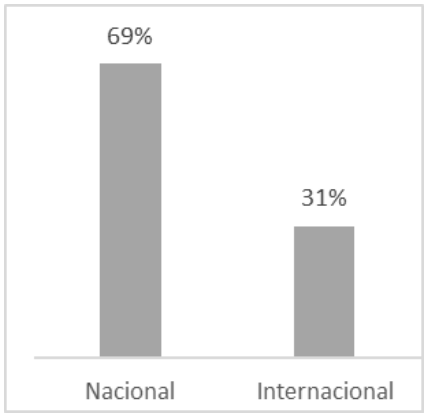

Figura 5.3:Atuação da empresa

A respeito do tamanho e quadro de funcionários das empresas participantes, 41% são empresas de grande porte, 31% de pequeno e 21% microempresa (Figura 5.5). Sobre a atividade primária desenvolvidas pelas empresas, aquelas que desenvolvem software sob encomenda se destaca com 41% (Figura5.6).

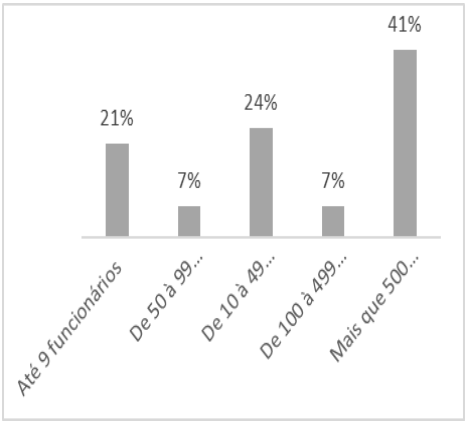

Figura 5.6: Quadro de funcionários e tamanho das empresas

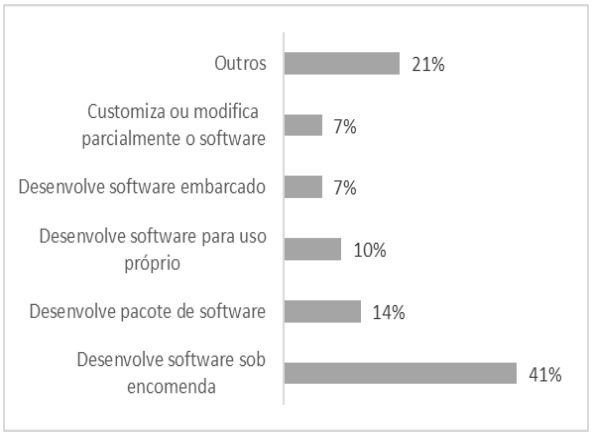

Figura 5.5: Atividade primária das empresas

Foi coletada também a informação sobre o perfil do respondente da pesquisa, no qual 28% das respostas foram feitas por analistas de sistemas seguido de 14% por diretores, 10% por implementadores e 7% por arquitetos de software (Figura 5.7).

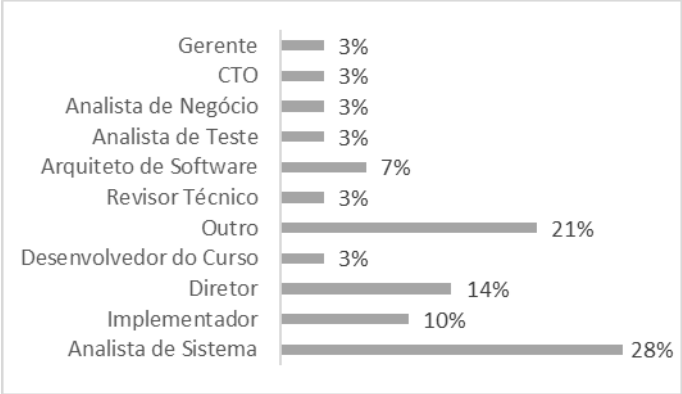

Figura 5.7 Perfil do respondente da pesquisa

A seguir, na próxima seção é abordado a utilização de modelos no desenvolvimento de software e sobre algumas transformações realizadas.

5.2 Modelos

A primeira questão abordada se deu a respeito do processo de levantamento de requisitos a fim de descobrir quais técnicas são utilizadas pelas empresas participantes da pesquisa, em que 19% das empresas utilizam entrevistas para a realização do levantamento de requisitos, 17% brainstorming, 13% questionários e 8% observação, como apresentado na Figura 5.8.

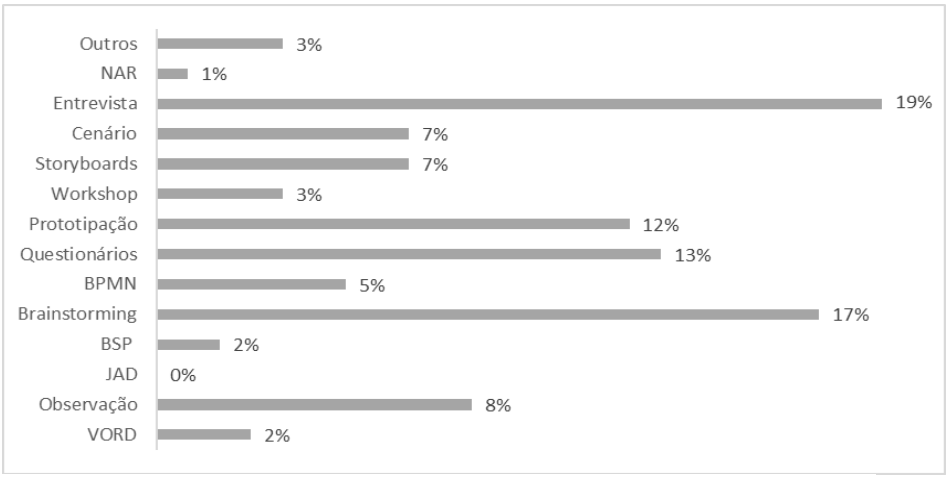

Figura 5.8: Técnicas utilizadas no levantamento de requisitos

Foi questionado também se o responsável por realizar o levantamento de requisitos é a mesma pessoa responsável por realizar a especificação dos requisitos; 72% das empresas afirmaram que sim e 28% disseram que não (Figura 5.10). Sobre o tipo de documento em que as especificações são armazenadas, 40% correspondem a documentos digitais, 23% a diagramas e 20% a modelos, conforme mostra a Figura 5.9.

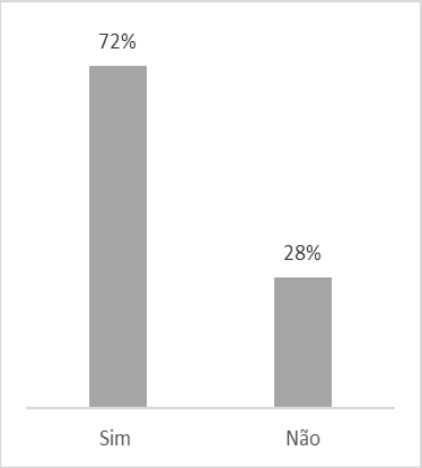

Figura 5.10: Responsável pelo levantamento e especificação dos requisitos

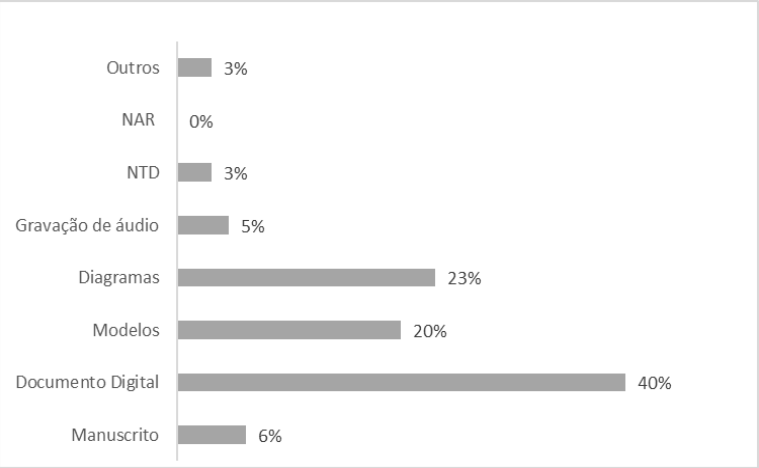

Figura 5.9: Tipo de documentos que são armazenados os requisitos

Foi identificado quais tipos de notações que as empresas utilizam para especificarem os requisitos e foi constatado que 28% utilizam histórias de usuários (HU), 22% modelos de caso de uso (MCU), 14% diagramas de atividade (DA) e 13% método formal (MF) (Figura 5.11).

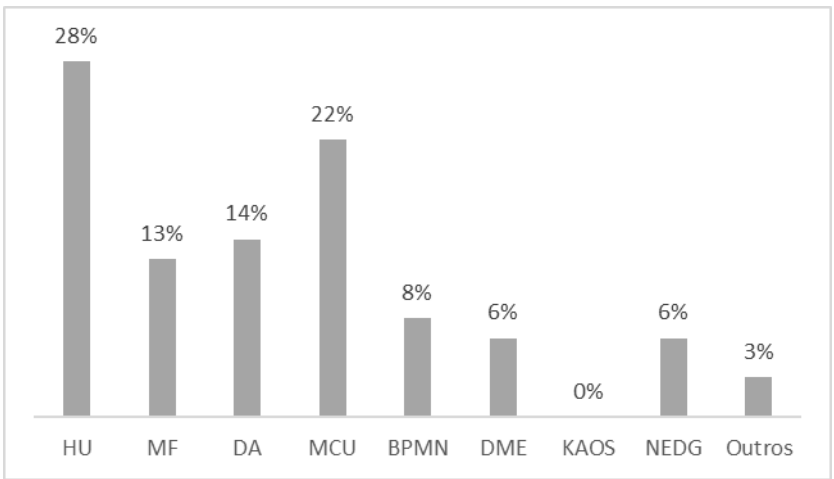

Figura 5.11: Notações utilizadas para especificações dos requisitos

Na fase de projeto de software, incialmente foi verificado quais são as representações utilizadas para representar a visão estrutural (estática) nas empresas. 22% responderam que utilizam diagramas de entidade relacionamento, 20% diagramas de classe (DCL), 12% diagramas de atividade e 20% afirmaram que não são utilizadas representações na empresa (NUE) (Figura 5.12).

Já para representar a visão comportamental, 30% utilizam de fluxogramas, 14% diagramas de atividade (DA), 10% diagramas de sequência (DS) seguido por 22% que não utilizam representações na empresa (NUE) (Figura 5.13).

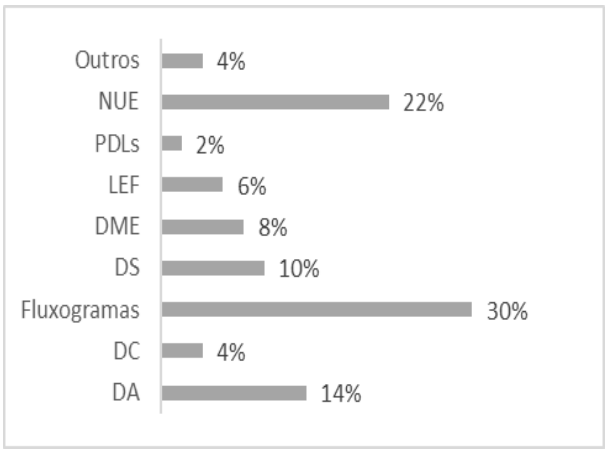

Figura 5.13: Representação visão comportamental

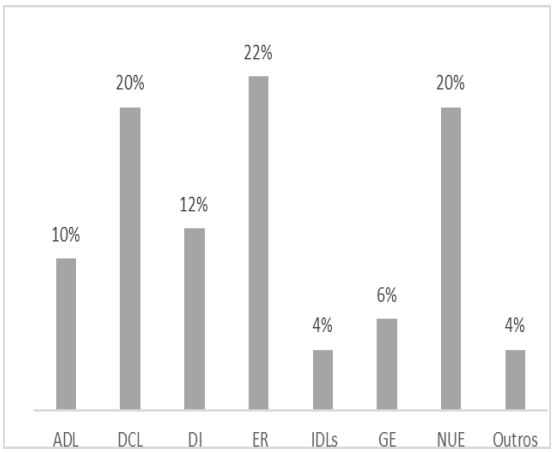

Figura 5.12: Representação visão estrutural

A especificação de um artefato pode ser utilizada para criação de outro artefato até mesmo de forma automática ou semiautomática por meio de uma ferramenta ou não. Na pesquisa foi apontado que 47% destas transformações são realizadas manualmente, 20% por meio de ferramenta semiautomática, 13% por meio de uma ferramenta automaticamente e 20% não realizam transformações (Figura 5.14).

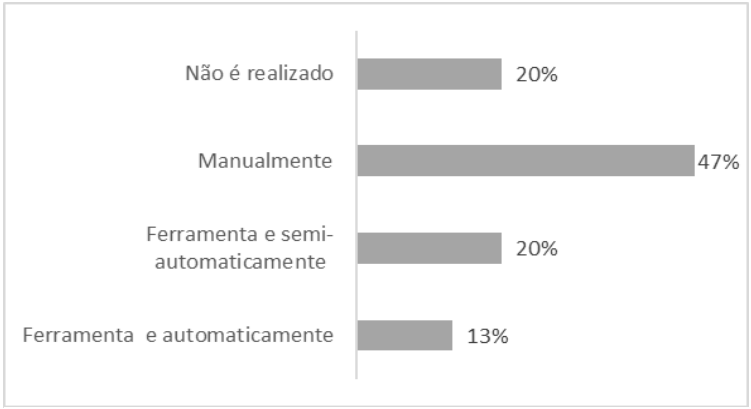

Figura 5.14:Método de transformação entre artefatos

Também foram analisados, na fase de projeto de software, a partir de quais artefatos (colunas) os artefatos estruturais (estáticos) (linhas) são criados. Conforme é apresentado na Tabela 5.1, os artefatos que se destacam e são utilizados como base para criação de outros artefatos são HU, MDU, DCL, DME e ER.

Tabela 5.1:Transformações entre os artefatos para representação estrutural

	HU	MF	MCU	DCL	DA	BPMN	DME	ER	KAOS	NG	Nenhum
DCL	14%	4%	14%	14%	6%	6%	4%	10%	0%	18%	10%
DI	7%	2%	11%	7%	9%	0%	9%	7%	0%	25%	23%
IDLs	8%	3%	10%	8%	8%	0%	0%	8%	0%	25%	33%
DC	11%	5%	5%	3%	3%	0%	8%	3%	0%	30%	32%
GE	13%	3%	3%	0%	6%	0%	0%	3%	0%	35%	35%
ER	15%	5%	13%	10%	3%	3%	3%	13%	0%	18%	18%

A seguir é apresentado um gráfico da distribuição dos dados contidos na (Tabela 5.1), onde é possível ver o alto índice apontando que GE não é utilizado em 35% nem gerado em 35% das empresas juntamente com DC que não é gerado em 30% e não é utilizado em 32%. Pode-se destacar também que os índices que saem da curva dos 10% e apresentam resultados mais satisfatórios são as transformações que acontecem entre: HU para DCL, HU para DC, HU para GE, HU para ER assim como MCU para DCL, MCU para DI, MCU para ER e ER para ER apresentado na Figura 5.15.

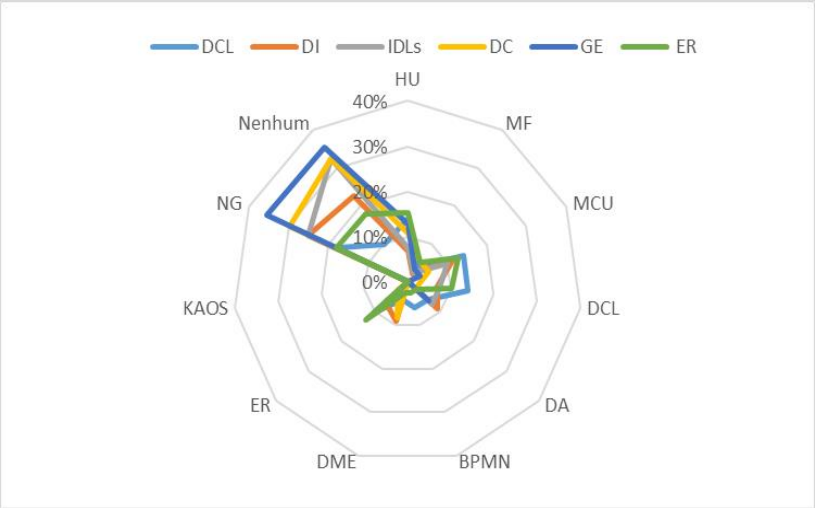

Figura 5.15: Representação gráfica das transformações entre os artefatos para representação estrutural

Na mesma linha da análise anterior, só que voltada para a representação comportamental da fase de projeto de software, foram identificadas as seguintes informações: uma média de 26% das empresas não geram os artefatos descritos e 31% geram os artefatos das colunas, mas não são transformados em outros artefatos. Já quanto a transformação entre os artefatos se destacam HU para DA 13%, HU para DME 11%, HU para Fluxogramas11%, MCU para DA 18%, MCU para DME 11%, MCU para Fluxograma 17%, DCL para fluxograma 13%, DA para DS 13% conforme apresentado na Tabela 5.2.

Tabela 5.2: Transformações entre os artefatos para representação estrutural

	HU	MF	MCU	DCL	DA	BPMN	DME	KAOS	NGA	Nenhum
DA	13%	5%	18%	8%	5%	0%	3%	0%	28%	23%
DS	3%	3%	10%	10%	13%	0%	5%	0%	28%	30%
DME	11%	3%	11%	5%	8%	0%	5%	0%	29%	29%
PDLs	6%	3%	9%	9%	3%	3%	0%	0%	29%	40%
Fluxogram	11%	4%	17%	13%	9%	7%	4%	0%	20%	15%
ER	9%	6%	9%	11%	9%	3%	0%	0%	20%	34%
LEF	7%	7%	3%	0%	3%	3%	0%	0%	33%	43%

Foi constatado também em quais situações os modelos normalmente são consultados pelos colaboradores. Com exceção das alternativas “outros” 1%, “nunca são consultados” 2% e dúvidas relacionadas ao gerenciamento do projeto 6%, os demais apresentados na Figura 5.16 ficaram entre os valores de 8% a 12%.

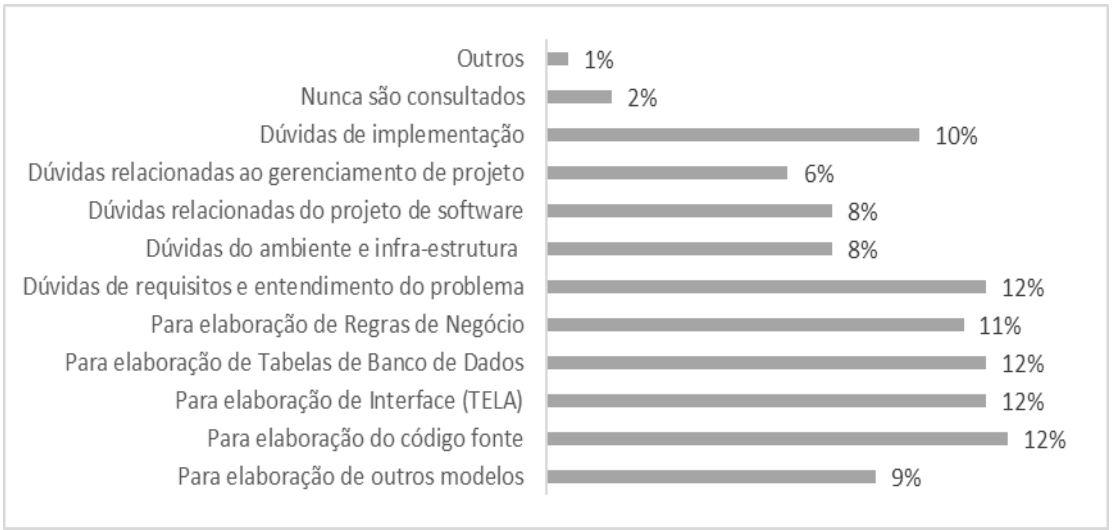

Figura 5.16: Motivo de consulta dos modelos pelos colaboradores

Também foi analisado o seguinte cenário: quando ocorrem mudanças no desenvolvimento do projeto, quais são os artefatos que são alterados para refletir as alterações necessárias. 41% das empresas afirmaram serem os artefatos de requisitos, 25% os artefatos de implementação, 24% os artefatos de projeto e em 10% afirmaram que não são realizadas alterações (Figura 5.17).

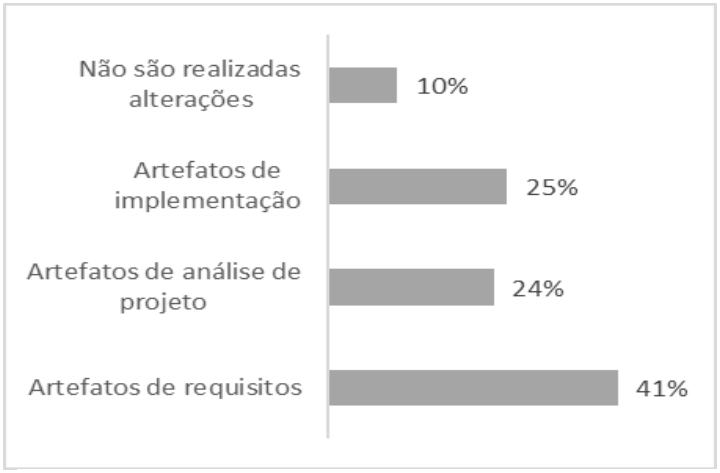

Figura 5.17: Alterações em artefatos

Após perguntado sobre a frequência e em quais etapas os projetos desenvolvidos na empresa são representados por modelos, obteve-se os seguintes valores: 28% na fase de requisitos ‘sempre’ são representados por modelos, enquanto 14% na fase de projeto e 7% na fase de construção. Quando são representados “quase sempre”, 21% na fase de requisitos, 28% no projeto, 28% na construção. Já na opção “maioria das vezes”, 28% na fase de requisitos, 31 % no projeto e na construção. Em relação aquelas que disseram “raramente” utilizarem a representação 14% na fase de requisitos, 14% na fase de projeto e 24 na construção. Enquanto aquelas que afirmaram “nunca” utilizarem, 10% na fase de requisitos, 14% no projeto e 10% na construção (Figura 5.18).

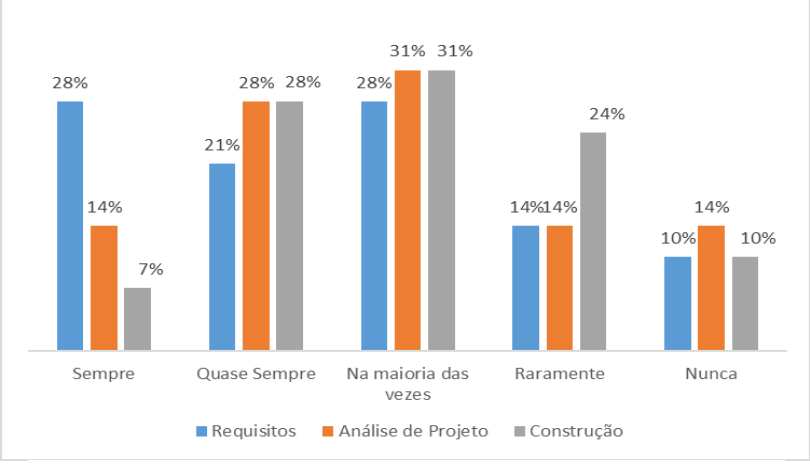

Figura 5.18: Frequência e representações por modelos

Na fase de construção também foi consultado sobre a ocorrência da transformação automática e semiautomática em códigos finais (códigos fontes, tabelas do banco de dados e interfaces gráficas) e a partir de quais artefatos eles podem ser gerados. As transformações que se destacam são: DCL para código fonte, DCL para geração de tabelas de banco de dados, DCL para CRUD, SCRIPT para geração de código fonte, SCRIPT para geração de tabelas de banco de dados e assim como outros apresentados na Tabela 5.3.

Tabela 5.3:Transformação automática e semiautomática para códigos finais

	DCL	SCRIPT	DC	DS	PROT	EMF	DI	NGA	NPO
Geração de código fonte	22%	19%	0%	5%	11%	0%	0%	27%	16%
Geração de tabelas banco de dados	17%	23%	3%	3%	3%	0%	0%	31%	20%
Geração de Interface (Tela)	0%	13%	3%	0%	19%	3%	0%	41%	22%
CRUD	19%	19%	3%	3%	3%	0%	3%	31%	19%
Regras de Negócio	10%	3%	0%	3%	7%	0%	0%	45%	31%
Apenas código fonte para um Framework específico	7%	7%	0%	3%	0%	0%	0%	41%	41%

Em seguida, as empresas que realizam transformações automáticas citaram alguns tipos de ferramentas que utilizam para realizar a transformação, algumas delas são: *Visual Studio Generator, XSL, XML, Bat, Astah, GeneXus, NetBeans, XMind, Dwins, Jira, Confluence, Enterprise Architect, Scripts de automação, Vscode, WebPack.*

5.3 Mapeamento

A primeira questão é sobre o quanto cada técnica da Tabela 5.4 favorece a interação do analista com o cliente para extração dos requisitos iniciais do sistema devido ao alto número de empresa que afirmaram a alternativa “não posso opinar” (NPO), foi realizada uma média para descobrir os que favorecem mais a interação. Destaque para brainstorming com média (4), questionários (3,7), prototipação (4,3), storyboards (4) e entrevista (4,3) apresentados na Tabela 5.4.

Tabela 5.4: Técnicas que favorecem no levantamento dos requisitos

	NPO	**1**	**2**	**3**	**4**	**5**	**Média**
ADLs	55%	0%	3%	24%	14%	3%	3.4
DCL	34%	3%	0%	7%	34%	21%	4.1
DI	55%	3%	0%	7%	28%	7%	3.8
ER	28%	0%	0%	14%	28%	31%	4.3
IDLs	55%	3%	3%	10%	14%	14%	3.7
GE	52%	0%	10%	21%	7%	10%	3.4

Já em relação as formas de documentações que mais favorecem o **entendimento** podem-se destacar três principais artefatos HU (4,3), MCU (4) e BPMN (3,9). Há também uma alta taxa de empresas que citam NPO conforme a Tabela 5.5.

Tabela 5.5: Tipos de documentos que favorecem no entendimento

	NPO	1	2	3	4	5	Média
VORD	59%	10%	7%	14%	7%	3%	2.7
Observação	31%	3%	14%	34%	7%	10%	3.1
JAD	83%	0%	3%	14%	0%	0%	2.8
BSP	69%	0%	7%	3%	14%	7%	3.7
Brainstorming	14%	3%	7%	10%	34%	31%	4
BPMN	52%	3%	3%	17%	17%	7%	3.5
Questionários	14%	3%	14%	10%	41%	17%	3.7
Prototipação	24%	0%	3%	7%	31%	34%	4.3
Workshop	41%	7%	3%	17%	10%	21%	3.6
Storyboards	28%	3%	3%	10%	34%	21%	4
Cenário	38%	10%	10%	3%	17%	21%	3.5
Entrevista	17%	0%	3%	14%	28%	38%	4.3

Já em relação as formas de documentações que mais favorecem na **descrição** podem-se destacar três principais artefatos HU (3,8), MCU (4) e BPMN (3,8). Há também uma alta taxa de empresas que afirmaram NPO, conforme mostra a Tabela 5.6.

Tabela 5.5: Tipos de documentos que favorecem na descrição/escrita

	NPO	1	2	3	4	5	Média
HU	28%	7%	7%	10%	24%	24%	3.8
MF	45%	10%	10%	17%	10%	7%	2.9
DA	38%	7%	7%	14%	21%	14%	3.5
MCU	34%	3%	0%	17%	21%	24%	4
BPMN	59%	3%	3%	10%	7%	17%	3.8
DME	55%	7%	10%	17%	7%	3%	2.8
KAOS	79%	3%	0%	3%	10%	3%	3.5

Foi abordado também dentro da fase de projeto de software quais representações de visão estrutural (estática) podem favorecer mais o **entendimento** para as fases posteriores do desenvolvimento, as três principais representações foram: DCL (4,1), DI (3,8) e ER (4,3), pode-se observar também as empresas que marcaram a alternativa NPO (Tabela 5.7).

Tabela 5-6: Representações estruturais que favorecem o entendimento

	NPO	1	2	3	4	5	Média
HU	21%	0%	7%	7%	28%	38%	4.3
MF	38%	3%	0%	21%	24%	14%	3.8
DA	34%	3%	10%	17%	21%	14%	3.5
MCU	28%	3%	0%	14%	34%	21%	4
BPMN	59%	3%	0%	10%	14%	14%	3.9
DME	59%	3%	0%	21%	14%	3%	3.4
KAOS	83%	3%	0%	3%	10%	0%	3.2

Também foi abordado na fase de projeto de software quais representações de visão estrutural (estática) podem favorecer mais a **descrição** para as fases posteriores do desenvolvimento. As três principais representações foram DCL (4,0), DI (3,9) e ER (3,9) levando em consideração as empresas que também marcaram a alternativa NPO (Tabela 5.8).

Tabela 5.7: Representações estruturais que favorecem a descrição

	NPO	1	2	3	4	5	Média
ADLs	52%	3%	10%	17%	3%	14%	3.3
DCL	38%	3%	0%	14%	21%	24%	4.0
DI	55%	3%	0%	7%	24%	10%	3.9
ER	28%	3%	3%	17%	24%	24%	3.9
IDLs	55%	0%	10%	3%	14%	17%	3.9
GE	52%	3%	10%	10%	17%	7%	3.3

Ainda na fase de projeto de software, mas relacionado ao quanto as representações de visão comportamental (dinâmica) podem favorecer o **entendimento** para as fases posteriores do desenvolvimento, as três principais representações destacadas são: DA (3,9), fluxograma (4,1) e DS (3,9) (Tabela 5.9).

Tabela 5.8 Representações compartimentais que favorecem o entendimento

	NPO	1	2	3	4	5	Média
DA	45%	0%	3%	14%	24%	14%	3.9
DC	48%	0%	7%	14%	24%	7%	3.6
Fluxograma	34%	0%	0%	17%	28%	21%	4.1
DS	38%	3%	3%	14%	21%	21%	3.9
DME	55%	3%	0%	17%	14%	10%	3.7
LEF	38%	0%	17%	17%	10%	17%	3.5
PDLs	52%	0%	7%	21%	10%	10%	3.5

Também na fase de projeto de software, relacionado ao quanto as representações de visão comportamental (dinâmica) podem favorecer a **descrição** para as fases posteriores do desenvolvimento, as três principais representações destacadas são: DA (4,2), fluxograma (4,3) e DS (3,9) (Tabela 5.10).

Tabela 5.9: Representações compartimentais que favorecem o entendimento

	NPO	1	2	3	4	5	Média
DA	45%	0%	0%	10%	24%	21%	4.2
DC	48%	0%	14%	21%	7%	10%	3.3
Fluxograma	34%	3%	0%	7%	24%	31%	4.3
DS	38%	7%	0%	10%	24%	21%	3.9
DME	55%	3%	3%	17%	7%	14%	3.6
LEF	38%	3%	14%	17%	14%	14%	3.4
PDLs	55%	3%	7%	14%	10%	10%	3.4

Quanto à geração automática ou semiautomática, total ou parcial sobre o nível de benefícios que podem trazer para o desenvolvimento de software de acordo com a pesquisa realizada a transformação favorece principalmente a padronização de código e também a produtividade conforme é apresentado na Tabela 5.11.

Tabela 5.10: Benefícios da transformação automática ou semiautomática de códigos finais.

	NPO	1	2	3	4	5	Média
Padronização de código	28%	0%	0%	17%	17%	38%	4.3
Facilidade de entender o código	28%	14%	3%	10%	17%	28%	3.6
Produtividade	28%	3%	0%	7%	24%	38%	4.3
Manutenção do código	28%	7%	10%	17%	17%	21%	3.5

Após apresentados os resultados da pesquisa na próxima seção é apresentada a discussão e as considerações finais.

6. DISCUSSÃO

A discussão da pesquisa tem como base as respostas das três perguntas objetivo do trabalho que são:

1. Quais modelos e técnicas são utilizados pela empresa no processo de desenvolvimento de software?
2. Dentro do processo de desenvolvimento de software existe transformação entre modelos e como são mapeados?
3. Como os modelos utilizados pelas empresas podem influenciar no processo de criação da aprendizagem organizacional?

A seguir serão discutidas cada uma das questões objetivas separadamente:

1. Quais modelos e técnicas são utilizados pela empresa no processo de desenvolvimento de software?

Para responder esta questão objetivo são utilizadas as sub questões do mapeamento, conforme apresentado no Quadro 4.2 na Seção 4.

- ***1.1- Quais modelos são mais comumente utilizados em cada fase (Requisito/Projeto)?***

Requisitos de software

Sobre a utilização de modelos na fase de levantamento de requisito é possível observar que 96% das empresas se apoiam em algum tipo de técnica. Este é um resultado muito bom para a engenharia de software, pois a etapa de requisitos é uma das mais importantes dentro do desenvolvimento, já que as etapas posteriores dependem dela.

No levantamento de requisitos a principal forma de obter os resultados é através da entrevista com o cliente, com o contato direto; seguido pelo brainstorming e questionários, as quais são apontadas por favorecer a interação com o cliente face a face.

Apesar das técnicas de levantamento serem antigas, a forma de realizar o armazenamento do que foi coletado durante o processo de interação com o usuário que se destaca é a por meio de documentos digitais. Dentro deste formato, tem-se as representações por diagramas 23% e modelos 20%, podendo-se afirmar novamente que são utilizados modelos na fase de requisitos de software.

Ainda na fase de requisitos de software, referente ao tipo de notação mais utilizada para realizar a descrição dos requisitos, as representações que tiveram resultados mais satisfatório foram: história de usuário e diagrama de caso de uso. Estas duas notações necessitam de uma abrangente interação com o usuário para serem construídas e as técnicas de entrevista, brainstorming e questionário possibilitam fortemente esta interação.

Projeto de software

Na representação estrutural do projeto, 20% das empresas não utilizam nenhuma representação, enquanto na representação comportamental 22% também não utilizam nenhuma representação.

Esses números são preocupantes, porém, foi afirmado por algumas empresas que a não utilização constante destes modelos nesta fase, se dá pelo emprego de métodos ágeis dentro da empresa, onde os desenvolvedores participam de reuniões constantemente com os analistas de requisitos tornando um processo iterativo incremental.

Das 80% das empresas que afirmaram que utilizam modelos para representar a visão estrutural do projeto, os dois modelos que se destacam são: modelo de entidade relacionamento e diagrama de classe. Estes modelos favorecem muito no seguinte aspecto: quando o desenvolvedor não teve acesso às etapas iniciais do projeto, mas precisam entender como está estruturado o projeto. Os modelos oferecem a valiosa informação de como se dá o relacionamento entre as classes e atributos dos objetos.

Da 78% das empresas que afirmaram que utilizam modelos para representar a visão comportamental do projeto, os modelos apontados como mais utilizados são: fluxogramas, diagrama de atividade e diagrama de sequência. O fluxograma foi o mais apontado dentre os modelos, isto se dá pela facilidade que se tem de construí-lo e também a facilidade de compreensão.

- ***1.2- Uma única pessoa é responsável por criar artefatos de etapas diferentes, existe uma avaliação dos modelos ou consulta para extrair soluções perante as dúvidas?***

Foi analisado na pesquisa que no processo de requisito, se a pessoa que é responsável por atuar no levantamento é a mesma responsável por realizar a descrição dos requisitos coletados. Constatou-se que em 78% das empresas é a mesma pessoa que realizam as atividades da fase de análise e projeto de software.

Pode-se relacionar ao fato de favorecer a similaridade entre o que o cliente deseja e a documentação que será utilizada nas fases posteriores do desenvolvimento.

Referente a qual(is) situação(ões) normalmente os modelos são consultados pelos colaboradores. Pode-se afirmar que as situações que mais necessitam ser consultadas de acordo com a pesquisa são: quando se tem dúvidas relacionadas ao "entendimento do problema" e "requisitos", para a "elaboração do código fonte", "elaboração de tabelas". São áreas que dependem fortemente das representações por meios de modelos e diagramas que influenciam em todo o desenvolvimento de software.

É possível observar também que quando necessitam ocorrer mudanças no projeto os artefatos que mais são alterados para satisfazer a mudança são os artefatos de requisitos, justamente aqueles que conforme apresentado anteriormente, obtendo-se maior número de dúvidas, o que fica evidente que os modelos são muito importantes no desenvolvimento de software principalmente na fase de requisitos.

- ***1.3- Qual a frequência que a empresa utiliza os modelos nas fases de desenvolvimento?***

Após coletados os dados dos resultados da pesquisa é possível observar que:

Na fase de requisitos: somando os valores de empresas que raramente utilizam modelos e aquelas que nunca utilizam, o valor chega aos 24%. 76% restantes são distribuídos da seguinte forma: 28% sempre utilizam os modelos, 21% quase sempre utilizam modelos e 28% na maioria das vezes utilizam modelos em seus projetos.

Na fase de projeto: somando os valores de empresas que raramente utilizam modelos e aquelas que nunca utilizam o valor chega a 28%. 72% restantes são distribuídos da seguinte forma:14% sempre utilizam os modelos, 28% quase sempre utilizam os modelos e 31% na maioria das vezes utilizam modelos em seus projetos.

Na fase de construção: somando os valores de empresas que raramente utilizam modelos e aquelas que não utilizam o valor chega a 34%. 66% restantes são distribuídos da seguinte forma: 7% sempre utilizam os modelos, 28% quase sempre utilizam os modelos e 31% na maioria das vezes utilizam modelos em seus projetos.

É visível que a maior frequência da utilização de modelos se dá na fase de requisitos, onde a porcentagem do valor das empresas que sempre utilizam modelos para realizar a representação é descrita como o dobro da fase de projeto de software e o triplo da fase de construção. Os resultados até aqui se assemelham em relação às respostas das questões abordadas anteriormente.

De um modo geral, respondendo à pergunta objetivo número 1, é possível observar que as empresas de desenvolvimento de software utilizam modelos em suas fases de desenvolvimento principalmente na fase de requisitos. Na fase de requisitos as técnicas mais utilizadas para o levantamento de requisitos são: entrevista, brainstorming e questionário. Quanto as notações utilizadas para representar as especificações de requisitos destaque para história de usuário e diagrama de caso de uso. Por fim, os documentos gerados são fortemente representados por modelos e diagramas em um formato digital.

As empresas utilizam também modelos na fase de projeto de software cerca de 10% a menos que na fase de requisitos. Dentre os modelos que são mais utilizados nesta fase são: diagrama de classe, diagrama de atividade, diagrama de entidade relacionamento, diagrama de sequência e fluxogramas.

2. **Dentro do processo de desenvolvimento de software existe transformação entre modelos e como são mapeados?**

- ***2.1. Como ocorre a transformação entre os modelos?***

Em 47% das empresas que responderam à pesquisa foi afirmado que a geração de um artefato para outro é realizada manualmente; 33% afirmaram que é realizado por meio de uma ferramenta semiautomática ou automaticamente; e 20% disseram que não é realizada a geração dos artefatos.

Das empresas que afirmaram que realizam a transformação de um modelo para outros modelos, semiautomático ou automaticamente, as seguintes ferramentas foram utilizadas: Visual Studio Generator, XSL, XML, Bat, Astah, GeneXus, NetBeans, XMind, Dwins, Jira, Confluence, Enterprise Architect, Scripts de automação, Vscode, WebPack.

- ***2.2. Quais artefatos são responsáveis por gerar outros artefatos?***

Relacionado ao projeto de software, buscou entender a partir de quais artefatos os artefatos comportamentais (dinâmicos) são criados. A soma das empresas que disseram que não usam ou que não é gerado o artefato descrito é de 57%. Dentre os 43% restante, os artefatos mais citados foram:

- *Fluxogramas:* os artefatos mais utilizados no seu processo de criação foram: modelo de caso de uso, história de usuário e diagrama de classe.
- *Diagrama de atividade:* os artefatos mais utilizados no seu processo de criação foram: modelo de caso de uso, história de usuário.
- *Diagrama de máquina e estado:* os artefatos mais utilizados no seu processo de criação foram: modelo de caso de uso, história de usuário e diagrama de atividade.

Ainda sobre o projeto de software, buscou entender a partir de quais artefatos os artefatos estruturais (estáticos) são criados. A soma das empresas que disseram que não usam ou que não é gerado o artefato descrito é de 50%. Dentre os 50% restante os artefatos que obtiveram resultados melhores foram os tais:

- *Diagrama de classe:* os artefatos mais utilizados no seu processo de criação foram: modelo de caso de uso, história de usuário e diagrama de entidade relacionamento.
- *Diagrama de entidade relacionamento:* os artefatos mais utilizados no seu processo de criação foram: modelo de caso de uso, história de usuário, diagrama de classe.
- *Diagrama de implementação:* os artefatos mais utilizados no seu processo de criação foram: modelo de caso de uso, diagrama de atividade e diagrama de máquina e estado.

- ***2.3. A utilização de modelos pode gerar código finais automáticos ou semi-automaticamente?***

Sobre a utilidade de modelos serem transformados em códigos finais (códigos fonte, tabelas de banco de dados e interfaces gráficas) de maneira automática ou semiautomática, 61% das empresas afirmaram não realizar a transformação ou então não podiam opinar sobre. 39% das empresas que disseram

que ocorre transformação, mostra a geração de código fonte como o maior resultado de transformações criados a partir de outros artefatos. A seguir é apresentado o processo de transformações dos mesmos.

- *Código fonte:* é criado principalmente por meio dos seguintes artefatos: diagrama de classe, script de automatização e prototipação.
- *Geração de tabelas do banco de dado:* transformado a partir de diagrama de classe, script de automatização.
- *CRUD:* transformado a partir de diagrama de classe e script de automatização.

- ***2.4. A geração automática de código contribui no desenvolvimento?***

Segundo o resultado da pesquisa a geração de código de forma automática pode favorecer principalmente a padronização do código fonte e a produtividade da equipe e o desenvolvedor. 28% das empresas que responderam à pesquisa disseram que não podiam opinar sobre o assunto.

Mesmo com um baixo número de transformações, algumas empresas adotam a transformações de modelos, a qual sua grande maioria é realizada manualmente. Em relação a transformação de código fonte total ou parcial, são realizados através de ferramentas e ajudam na padronização do código e na produtividade do desenvolvimento.

3. Como os modelos utilizados pelas empresas podem influenciar no processo de criação da aprendizagem organizacional?

- ***3.1. Como e quais os modelos favorecem o ensino e aprendizagem no processo de socialização?***

Socialização: Nonaka e Konno (1998) definem a socialização, como o processo de conversão do conhecimento tácito de uma pessoa para o conhecimento tácito de outra pessoa, geralmente esse conhecimento é adquirido através do compartilhamento de experiências, por meio de encontros, reuniões, entrevistas.

O conhecimento tácito de uma pessoa pode ser definido como aquele conhecimento que é adquirido com o tempo através de trocas de experiência, é um conhecimento que é mais difícil de ser formalizado.

Na etapa de análise de requisitos, principalmente na etapa de levantamento de requisitos, a qual o analista vai até o cliente, convive com as suas experiências,

conhece como é o processo que deve ser implementado na aplicação, conhece também as dores do cliente. Por meio das experiências com o cliente, visando extrair o máximo de informações para auxiliar nas demais etapas do desenvolvimento de software.

De acordo com o resultado da seguinte questão: avalie as técnicas considerando quanto cada uma favorece a interação do analista com o cliente para extração dos requisitos iniciais do sistema?

Dentre as técnicas presentes nas alternativas a “entrevista” foi a segunda técnica mais bem selecionada da questão.

Pode-se atribuir a entrevista como um processo de socialização assim como citado por Nonaka e Kono (1998), pois é a partir dela que ocorre o processo de troca de experiências, de um lado o cliente que conhece sobre o seu processo mas não consegue formaliza-lo em modelos ou representações seu conhecimento (tácito) e por outro lado o analista de sistema que não conhece o processo do cliente e através da entrevista começar a perceber o ambiente de desenvolvimento através da experienciação (tácito).

Também pode-se afirmar que no processo de socialização existe a presença dos modelos CIM, Sommerville (2007) afirma que é na fase de análise de requisitos que acontece as trocas de experiências entre cliente e desenvolvedor.

Desta forma os modelos independentes podem ser encontrados nas etapas iniciais de desenvolvimento, da quais são especificados detalhes da conversa entre o clientes e desenvolvedores, onde são expressados e descritos os requisitos do projeto, ou seja, no levantamento de requisitos.

- ***3.2. Como os modelos favorecem o ensino e aprendizagem no processo de externalização?***

Externalização: é considerada a transformação do conhecimento tácito em conhecimento explícito, ou seja, o indivíduo assimila aquilo que é inato para ele e converte em uma forma explícita (NONAKA; KONNO, 1998) da qual pode ser representado por modelos, desenhos, escrita, sendo assim para que as demais pessoas que desconhecem o tema abordado, a partir desta demonstração ou representação passem a entender.

As seguintes questões abaixo são utilizadas para analisar a influência dos modelos CIM e PIM na aprendizagem organizacional no processo de externalização:

- ***3.2.1-*** *Avalie as formas de documentação abaixo, como cada uma favorece a descrição dos requisitos?*

Dentre as os modelos que favorecem mais a descrição/escrita dos requisitos estão: modelo de caso de uso (CIM), história de usuário (CIM) e BPMN (CIM). São os modelos os quais os analistas conseguem transferirem melhor o seu conhecimento tácito adquirido na etapa de levantamento de requisitos para um documento (conhecimento explícito).

- ***3.2.2-*** *Dentre as representações da visão estrutural (estática), avalie quanto cada uma favorece a descrição das informações necessárias para as fases posteriores do desenvolvimento?*

Já na fase de projeto quando se fala na descrição dos modelos para representar a visão estrutural. Foi apontado nos resultados da pesquisa que os diagramas de classe (PIM) é o melhor tipo de modelo para descrever um conhecimento obtido na etapa de requisitos, seja ela diretamente com o cliente ou através de reuniões realizadas com os responsáveis pelo levantamento dos requisitos.

- ***3.2.2-*** *Dentre as representações visão comportamental (dinâmica), avalie quanto cada uma favorece a descrição das informações necessárias para as fases posteriores do desenvolvimento?*

Quanto a representação da visão comportamental, os modelos que mais favorecem a descrição das informações apontado no resultado da pesquisa são: fluxograma (CIM/PIM) e o diagrama de atividade (CIM/PIM). O cenário relacionado com a aprendizagem organizacional abordado nessa questão é o seguinte: quando o analista ou até mesmo o engenheiro de software que participou do levantamento de requisito (conhecimento tácito) e na sequencia tem a tarefa de criar os modelos da representação comportamental a partir dos requisitos captados (conhecimento explícito).

É possível atrelar a externalização aos modelos PIM, pelo seguinte critério: quando são coletados requisitos na fase de análise, se faz necessário serem descritos de uma maneira que seja compreensiva para os engenheiros de software que são os responsáveis por arquitetar os modelos da fase de design tanto comportamentais como estruturais.

Desta forma é necessário que o analista que teve o contato inicial com o cliente, após passar pelo processo de socialização, descreva as especificações do sistema de uma forma legitima para os demais membros da equipe possam compreender os detalhes da implementação.

A externalização ocorre no momento em que o analista vai descrever a especificação dos requisitos, pois é de responsabilidade dele transmitir o conhecimento adquirido por meio das técnicas de requisitos (socialização) que serão utilizados nas próximas etapas do desenvolvimento por outros profissionais (externalização).

3.3. Como os modelos favorecem o ensino e aprendizagem no processo de combinação?

As seguintes questões abaixo são utilizadas para analisar a influência dos modelos CIM e PIM na aprendizagem organizacional no processo de combinação

- ***3.3.1-*** *Avalie as formas de documentação abaixo, como cada uma favorece o entendimento e descrição dos requisitos?*

Os melhores modelos avaliados que colaboram no entendimento dos requisitos são: história de usuário (CIM), modelo de caso de uso (CIM) e BPMN (CIM) já na descrição do entendimento dos requisitos são: fluxograma (PIM), diagrama de atividade (CIM) e diagrama de sequência (PIM).

- ***3.3.2-*** *Dentre as representações da visão estrutural (estática), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento?*

Na fase de projeto de software os modelos mais bem avaliados da visão estrutural quanto se tratam do entendimento foram: diagrama de classe (PIM), diagrama de entidade relacionamento (PSM) e diagrama de implementação (PSM) enquanto na descrição na descrição: diagrama de classe (PIM).

- ***3.3.3-*** *Dentre as representações visão comportamental (dinâmica), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento*?

Na fase de projeto de software os modelos mais bem avaliados da visão comportamental quanto se tratam do entendimento foram: fluxograma (PIM), diagrama de sequência (DS) e diagrama de atividade (CIM) e também na descrição: fluxograma (PIM), diagrama de sequência (DS) e diagrama de atividade (CIM).

As três questões respondidas anteriormente tratam do seguinte cenário: o engenheiro de software ou o responsável pela criação dos artefatos que não possuem conhecimento sobre o que aconteceu na fase de requisitos e necessita fazer a leitura em busca do entendimento dos modelos gerados da fase anterior (explícito) para que em seguida consiga realizar a criação/descrição dos artefatos

das fases posteriores do desenvolvimento (explícito), transformando assim a transformação do conhecimento explícito para conhecimento explícito (combinação).

Combinação: Nonaka e Konno (1998), definem a combinação como um processo de transformação do conhecimento explícito para o conhecimento explícito, ou seja, o indivíduo observa um modelo, uma estrutura escrita, e transforma em um modelo diferente: pode-se expressar na seguinte expressão: modelo resultante = modelo antigo + transformação. Isto ocorre principalmente na transformação de modelos CIM para PIM, e também de PIM para PIM. Pode-se citar um exemplo da transformação entre os diagramas comportamentais, um diagrama de caso de uso para um diagrama de atividade por exemplo.

- ***3.4. Como os modelos favorecem o ensino e aprendizagem no processo de internalização?***

Internalização: é considerado processo de transformação do conhecimento explícito para conhecimento tácito, o qual o indivíduo não tem conhecimento sobre determinado assunto ou tarefa (NONAKA; KONNO, 1998), a partir do momento em que ele faz contato com conhecimento explícito ele assimila e aplica também o seu conhecimento inato tácito para realizar determinada tarefa, fazendo a junção desses conhecimentos.

- ***3.4.1-*** *Avalie as formas de documentação abaixo, como cada uma favorece o entendimento dos requisitos?*

Os modelos que mais contribuem para o entendimento dos requisitos são: história de usuário (CIM), modelo de caso de uso (CIM) e BPMN (CIM).

- ***3.4.2-*** *Dentre as representações da visão estrutural (estática), avalie quanto cada uma favorece o entendimento das informações necessárias para as fases posteriores do desenvolvimento?*

Os modelos da etapa de projeto de software na representação estrutural que mais contribuem para o entendimento são: diagramas de entidade relacionamento (PSM), diagrama de classe (PIM) e diagrama de implantação (PSM).

- ***3.4.3-*** *Dentre as representações visão comportamental (dinâmica), avalie quanto cada uma favorece o entendimento das informações necessárias para as fases posteriores do desenvolvimento?*

Os modelos da etapa de projeto de software na representação comportamental que mais contribuem para o entendimento são: fluxograma (PIM), diagrama de atividade (CIM) e diagrama de sequência (PIM).

O processo de internalização é a transformação do conhecimento explícito para o conhecimento tácito, pode-se atribuir a relação entre as três respostas das questões anteriores ao processo de internalização. Pois quando se tem um determinado modelo, que está descrito formalmente (conhecimento explícito) e a partir da sua leitura/entendimento assimilando o conteúdo e logo após transformando aquele que era explicito para algo que é inato/tácito da pessoa que está realizando a leitura do modelo e dá a capacidade de realizar a combinação do conhecimento adquirido com seu conhecimento inato na criação de novos modelos.

Portanto é possui observar que os modelos que compõe o MDE podem influenciar sim na aprendizagem organizacional em especifico na construção do conhecimento dentro das empresas, desde o processo de socialização com os modelos da etapa de análise de requisitos até a internalização por meios dos modelos de requisitos e de projeto de software.

7. CONSIDERAÇÕES FINAIS

7.1 Contribuições

A primeira contribuição deste trabalho se dá pelo objetivo inicial que é analisar como o MDE compostos pelos seus modelos influenciam no processo de criação do conhecimento da aprendizagem organizacional. Oferecendo assim informações valiosas sobre quais técnicas e modelos que estão sendo mais utilizadas no desenvolvimento de software (requisitos, projeto e construção) em empresas brasileiras desenvolvedoras de software no cenário atual.

Outra contribuição do trabalho se dá pela realização do mapeamento dos principais modelos utilizados em cada fase do desenvolvimento, que foram utilizados no processo de criação do instrumento de pesquisa e a seguir aplicado nas empresas.

Foi visto que os modelos propostos na revisão bibliográfica são utilizados pelas empresas, apesar baixo número de transformações que são realizados estas são feitas de forma manual. Algumas empresas também utilizam os modelos para a geração de código fonte ou tabelas de banco de dados de forma automática ou semiautomática.

Outra contribuição do trabalho se trata da coleta e armazenamento das informações de contatos das empresas e principalmente a coleta e mapeamento dos funcionários através da rede social “Linkedin” criando uma rede de contatos, que pode facilitar a comunicação em outras futuras pesquisas.

7.2 Limitações

Uma das limitações que pode ser apontada no trabalho é a baixa quantidade de empresas respondentes.

Outra limitação do trabalho é o perfil dos entrevistados, que são pessoas que desempenham papéis diferentes dentro das empresas correndo o risco de não conseguirem opinar sobre algumas perguntas que não estavam dentro do seu campo de atuação na organização.

7.3 Trabalhos futuros

Como trabalhos futuros tem-se a possibilidade de buscar o aprimoramento do entendimento de como as novas tecnologias, modelos e ferramentas discutidos no trabalho estão relacionados com às teorias. Pretende-se realizar um estudo descritivo mais aprofundado, baseado em entrevistas pessoais ou observação dentro das empresas de desenvolvimento de software.

Além de abordar como os modelos e ferramentas se relacionam com a aprendizagem organizacional, pretende-se também pesquisar com mais detalhes os fatores culturais e organizacionais que influenciam o uso e a influência dos modelos com suas respectivas ferramentas.

Por fim, a ideia é direcionar a pesquisa para um cargo especifico dentro da organização que tenha conhecimento do ciclo do software por completo.

REFERÊNCIAS

ABES, Software, "**Mercado Brasileiro de Software: Panorama e Tendências**", Associação Brasileira das Empresas de Software; São Paulo SP; 1 edição, Junho 2018.

ARGYRIS, C.; SCHÖN, D. **Organizational Learning II: Theory, Method, and Practice**. Reading, MA, USA Cambridge, Uk: Addison Wesley, 1996.

AHMED, Abdelgaffar Hamed; SIDAHMED, Abeer Abd Allah; ELTOUM, Rehab Bakheet. Automation of test scripts in software product line using Model driven architecture. In: **Computing, Control, Networking, Electronics and Embedded Systems Engineering (ICCNEEE), 2015 International Conference on**. IEEE, 2015. p. 62-66.

BARBOSA, Daniel Mendes; BAX, Marcello. A Design Science como metodologia para a criação de um modelo de gestão da informação para o contexto da avaliação de cursos de graduação. **Revista Ibero-Americana de Ciência da Informação**, v. 10, n. 1, p. 32-48, 2017.

BELIX, José Eduardo. **Um estudo sobre MDA: suporte fornecido pela UML e reuso de soluções pré-definidas**. 2006. Tese de Doutorado. Universidade de São Paulo.

BENNETT, Jeannette; COOPER, Kendra; DAI, Lirong. Aspect-oriented model-driven skeleton code generation: A graph-based transformation approach. **Science of Computer Programming**, v. 75, n. 8, p. 689-725, 2010.

BOGDAN, Robert; BIKLEN, Sari. Investigação qualitativa em Educação: fundamentos, métodos e técnicas. **Investigação qualitativa em educação. Portugal: Porto Editora**, p. 15-80, 1994.

BØRNSON, F.; DINGSØYR, T. **Knowledge management in software engineering : A systematic review of studied concepts , findings and research methods used**. Information and Software Technology, v. 50, n. 11, p. 1055-1068, 2008.

BOURQUE, Pierre et al. **Guide to the software engineering body of knowledge (SWEBOK (R)): Version 3.0**. IEEE Computer Society Press, 2014.

BRAMBILLA, M; CABOT, JORDI; WIMMER, M. **Model-Driven Software Engineering in Practice.** Morgan & Claypool, 2012. 165p.

BROWN, Alan W.; CONALLEN, Jim; TROPEANO, Dave. Practical insights into model-driven architecture: Lessons from the design and use of an MDA toolkit. In: **Model-Driven Software Development**. Springer, Berlin, Heidelberg, 2005. p. 403-431.

CALIARI, Giuliano Luz Pigatti. **Transformações e mapeamentos da MDA e sua implementação em três ferramentas**. 2007. Tese de Doutorado. Universidade de São Paulo.

CARNEVALLI, José Antonio; MIGUEL, Paulo Augusto Cauchick. Desenvolvimento da pesquisa de campo, amostra e questionário para realização de um estudo tipo survey sobre a aplicação do QFD no Brasil. **21º Encontro Nacional de Engenharia de Produção**, p. 17-19, 2001.

CHAVES, Rafael Alves. **Aspectos e MDA: criando modelos executáveis baseados em aspectos.** 2004. 87 f. Dissertação (Mestrado) - Curso de Ciência da Computação, Programa de Pós-graduação em Ciência da Computação., Universidade Federal de Santa Catarina, Florianópolis, Sc, 2004.

CONFERENCE ON SYSTEMS SCIENCES, 31., 1998, Hawaii. **Anais...**, 1998,

DIAS, Felipe et al. Uma Abordagem para a Transformação Automática do Modelo de Negócio em Modelo de Requisitos. In: **WER**. 2006. p. 51-60.

FRANCE, R. B.; RUMPE, B. Model-driven Development of Complex Software: A Research Roadmap. In: **IEEE Future of Software Engineering**, 2007. FOSE '07, 37–54. doi:10.1109/FOSE.2007.14

FREITAS, Henrique et al. O método de pesquisa survey. **Revista de Administração da Universidade de São Paulo**, v. 35, n. 3, 2000.

GIL, Antonio Carlos. Como elaborar projetos de pesquisa. **São Paulo**, v. 5, n. 61, p. 16-17, 2002.

HUSSAIN, Ambreen; WU, Wenyan. Sustainable Interoperability and Data Integration for the IoT-Based Information Systems. In: **Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), 2017 IEEE International Conference on**. IEEE, 2017. p. 824-829.

JIANG, Ke; ZHANG, Lei; MIYAKE, Shigeru. OCL4X: An Action Semantics Language for UML Model Execution. In: **Computer Software and Applications Conference, 2007. COMPSAC 2007. 31st Annual International**. IEEE, 2007. p. 633-636.

KLEPPE, Anneke G. et al. The model driven architecture: practice and promise. 2003.

KOLB, D. **Experiential Learning: Experience as the Source of Learning and Development.** NJ USA: Prentice Hall, 1984.

LEVINE, L; MONARCH, I. **Collaborative Technology in the Learning Organization: Integrating Process with Information Flow, Access, and Interpretation.** In: IEEE INTERNATIONAL

MAFRA, Sômulo Nogueira; TRAVASSOS, Guilherme Horta. Estudos Primários e Secundários apoiando a busca por Evidência em Engenharia de Software. **Relatório Técnico, RT-ES**, v. 687, n. 06, 2006.

MAGALHAES, P. F.; DAVID, J.; MACIEL, R. S. P.; DA SILVA, F.A. Modden: An Integrated Approach for **Model Driven Development and Software Product Line Processes**. In: Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS 2011), 5., 2011, São Paulo, SP.Anais..., 2011, p. 21-30.

MELO, Luan de Souza. **Uma Abordagem para Geração de Código de Persistência de dados Hibernate baseado Em MDD e POA.** 2016. 89 f. TCC (Graduação) - Curso de Ciência da Computação, Centro de Ciências Tecnologicas, Universidade Estadual do Norte do Paraná, Bandeirantes, 2016.

MENOLLI, André Luís Andrade. **Ambiente colaborativo semântico voltado à Aprendizagem Organizacional para empresas de Desenvolvimento de Software**. 2013. 275 f. Tese (Doutorado) - Curso de Informática, Programa de Pós-Graduação em Informática, Pontifícia Universidade Católica do Paraná, Curitiba, 2013.

NAUR, P.; RANDELL, B. **Software Engineering**: Report on a Conference sponsored by the NATO Science Committee, Garmisch, Germany, 1969.

NEVIS, E. C.; DI BELLA, A.; GOULD, J. M**. Understanding organizations as learning systems**. Sloan Management Review, v. 36, n.2, p. 73-85, 1995.

NONAKA, I. **Gestão do Conhecimento.** Porto Alegre: Bookman, 2008, 319 p.

NONAKA, I.; KONNO, N. **The Concept of "Ba": Building a Foundation for Knowledge Creation**, California Management Review, Usa, v. 40, n. 3, p. 40-54, 1998.

OLIVEIRA, Valter Castelhano de. **Proposta de método para gestão de requisitos de sistemas integrando modelagem de negócio e linguagens formais**. 2008. Tese de Doutorado. Universidade de São Paulo.
p. 444-461.

PALUDO, Marco Antônio**. Reúso de software com ênfase em abordagens dirigidas a modelos e sistemas de alta variabilidade: Estudo de caso no brasil.** 2016. 330 f. Tese (Doutorado) - Curso de Programa Pós-Graduação em Informática, Pontifícia Universidade Católica do Paraná, Curitiba, 2016.

RECH, J.; BUNSE, C. **Model-Driven Software Development**: Integrating Quality Assurance. Hershey: Information Science Reference. 526p.

REZENDE, Denis Alcides. **Engenharia de software e sistemas de informação**. Brasport, 2005.

SENGE, P.; KLEINER, A.; ROBERTS, C.; ROSS, R.; SMITH, B. J. **The fifth discipline field book**. New York: Doubleday, 1994.

SCHMITZ, Carsten et al. LimeSurvey: An open source survey tool. **LimeSurvey Project Hamburg, Germany. URL http://www. limesurvey. org**, 2018.

SOMMERVILLE, I. **Software Engineering**. 9. ed. Boston: Addison-Wesley Publishing Company, 2011. 773 p.

SOUZA, Y. S. de. **Organizações de Aprendizagem ou Aprendizagem Organizacional**. **Rae-eletrônica**, v. 1, n. 3, p.1-16, 2004.

STAHL, T; VÖELTER, M. **Model-Driven Software Development**: Technology, Engineering, Management. Chichester: John Wiley & Sons Ltd, 2006, 428p.

STANDISH GROUP. Estados Unidos. **The True Cost of a Project**. Disponível on-line em: <http://www.standishgroup.com/>. Acesso em 05 abr. 2018.

STEPHAN, Matthew; CORDY, James R. Model-driven evaluation of software architecture quality using model clone detection. In: **Software Quality, Reliability and Security (QRS), 2016 IEEE International Conference on**. IEEE, 2016. p. 92-99.

TAKEISHI, **A. Knowledge Partitioning in the Interfirm Division of Labor: The Case of Automotive Product Development.** Organization Science, v. 13, n. 3, p. 321-338, 2002.

TEECE, D. J; PISANO, G.; SHUEN, A. **Dynamic Capabilities and Strategic Management**. Strategic Management Journal, v.18, n.7, p. 509-533, 1997.

WENGER, E. **Communities of Practice:** Learning, Meaning, and Identity. Cambridge, Uk: Cambridge University Press, 1998.

APÊNDICE A – QUESTIONÁRIO

Titulo: Pesquisa do uso da engenharia dirigida à modelo no desenvolvimento de software.

1.Caracterização da Empresa

1.1 Estado_____

1.2Cidade____

1.3 Nome da Empresa_____

1.4 Ano de Fundação da Empresa_____

1.5 Início das Atividades em TI_____

1.6 Cargo do respondente da pesquisa * *(RUP, 2007)*

- Administrador de Sistemas
- Analista de Sistema
- Analista de Teste
- Analista de Processo e Negócio
- Analista de Negócio
- Arquiteto de Software
- Arquiteto de Segurança
- Artista Gráfico
- Coordenador de Projeto
- Coordenador de Revisão
- Desenvolvedor do Curso
- Designer
- Designer de Banco deDados
- Designer de Interface com o Usuário
- Designer de Negócios
- Designer de Teste
- Engenheiro de Processos
- Especialista em Ferramentas
- Especificador de Requisitos
- Gerenciador de Configuração
- Gerenciador de Controle de Mudanças
- Gerenciador de Implantação
- Gerenciador de Teste
- Implementador
- Integrador
- Redator Técnico
- Revisor
- Revisor de Gerenciamento

- Revisor Técnico
- Testador
- Outro: Especifique _____

1.7 O capital da sua organização é predominantemente: * *Adaptado de (MCT, 2009)*

- Privado
- Público

1.8 Tipo da empresa: * *Adaptado de (MCT, 2009)*

- Nacional
- Internacional

1.9 Atende os clientes * *Adaptado de (MCT, 2009)*

- Apenas Nacional
- América Do Sul
- América do Norte
- Europa
- Ásia
- África
- Oceania

1.10 Atividade Primária da Empresa é * *Adaptado de (MCT, 2009)*

- Desenvolve software para uso próprio
- Desenvolve software embarcado
- Desenvolve software sob encomenda
- Desenvolve pacote de software
- Customiza ou desenvolve parcialmente software
- Outra. Especifique _____

1.11 Tamanho da Empresa: * *(SEBRAE, 2012)*

- Até 9 funcionários
- De 10 a 49 funcionários
- De 50 a 99 funcionários
- De 100 a 499 funcionários
- Mais que 500 funcionários

2. Modelos

2.1 De que forma é realizada a interação com o cliente na fase de levantamento dos resquisitos de software?* *Bourque et al (2004)*

Marque todas que se aplicam.

- VORD (Orientada a ponto de vista)
- Observação
- Análise de Protocolo
- JAD (Joint Application Design)
- BSP (Business System Planning)
- BPMN(Business process Modeling Notation)
- Brainstorming
- Questionários
- Prototipação
- Workshop
- Storyboards
- Cenário
- Entrevista
- A empresa não atua na fase de levantamento de requisitos de software
- Outro. Especifique _____

2.2. O responsável por fazer a elicitação de requisitos, é o mesmo encarregado por fazer a especificação dos requisitos de software? *

Marcar apenas uma oval.

- Sim
- Não
- Outro. Especifique _____

2.3. Após a conversa inicial com o cliente, em que tipo de documento as especificações dos requisitos são armazenadas? * *Bourque et al (2004)*

Marque todas que se aplicam.

- Manuscrito
- Documento Digital

- Modelos
- Diagramas
- Gravação de áudio
- Não utiliza nenhum tipo de documento
- A empresa não atual na fase de requisitos
- Outro: Especifique _____

2.4. Qual tipo de notação a empresa utiliza na especificação dos requisitos? * *Adptado Bourque et al (2004).*

Marque todas que se aplicam.

- História de Usuário
- Método Formal
- Diagrama de Atividade
- Modelo de De Caso de Uso
- BPMN (Business Process Modeling Notation)
- Diagrama de Máquina de Estado
- Notação Kaos (Knowledge Acquisition in automated Specification)
- A empresa não especifica os documentos gerados
- Outro: Especifique _____

2.5. Dentre as representações existentes dos artefatos de projeto de software, quais são utilizadas para representar a visão estrutural (estática) na empresa?* *Adptado Bourque et al (2004).*

Marque todas que se aplicam.

- Linguagens de descrição de arquitetura (ADLs): linguagens textuais.
- Diagramas de classes.
- Diagramas de implantação.
- Diagramas de entidade relacionamento.
- Linguagens de descrição de interface (IDLs): linguagens de programação usadas para definir as interfaces.
- Gráficos de estrutura.
- Nenhuma representação é utilizada pela empresa.
- Outro: Especifique _____

2.6. Dentre as representações existentes dos artefatos da análise de projeto, quais são utilizadas para representar a visão comportamental (dinâmica) na empresa? * *Adptado Bourque et al (2004).*

Marque todas que se aplicam.

- Diagramas de atividade.
- Diagramas de comunicação.
- Fluxogramas.
- Diagramas de sequência.
- Diagramas de estado.
- Linguagens de especificação formal.
- Pseudocódigo e linguagens de design de programa (PDLs).
- Nenhuma representação é utilizada pela empresa.
- Outro: Especifique_____

**2.7. Quando um artefato é utilizado para gerar outro artefato ou código, de que forma isto é realizado?? ****Adptado Bourque et al (2004).*

Marque todas que se aplicam.

- Por meio de uma ferramenta automaticamente.
- Por meio de uma ferramenta semi-automaticamente.
- Manualmente.
- Não é realizado.
- Outro: Especifique_____

2.8. Na fase de análise de projeto assinale a partir de quais artefatos (colunas) os artefatos estruturais (estáticos) (linhas) são criados.

	Histórias de Usuário	Método Formal	Modelo de Caso de Uso	Diagrama de Classe	Diagrama de Atividade	BPMN	Máquina de Estado	Entidade Relacionamento	Notação Kaos	Não é gerado este artefato na empresa	Nenhum
Diagrama de Classe	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Diagrama de Implementação	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Linguagens de descrição de interface (IDLs)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Diagrama de Comunicação	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Gráfico de estrutura	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Diagramas de relação de entidade (ERDs)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

2.9. Na fase de análise de projeto assinale a partir de quais artefatos (colunas) os artefatos comportamentais (dinâmicos) (linhas) são criados.*

	Histórias de Usuário	Método Formal	Modelo de Caso de Uso	Diagrama de Classe	Diagrama de Atividade	BPMN	Diagrama de Máquina de Estado	Notação Kaos	Não é gerado este artefato na empresa	Nenhum
Diagrama de Atividade	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Diagrama de Sequência	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Diagrama de Estado	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Pseudocódigo e linguagens de design de programa (PDLs)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Fluxogramas	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Diagramas de relação de entidade (ERDs)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Linguagens de especificação formal	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

2.10. Em qual(is) situação(ões) normalmente os modelos são consultados pelos colaboradores? *

Marque todas que se aplicam.

- ○ Para elaboração de outros modelos
- ○ Para elaboração do código fonte (CRUD, código final, domínio e etc)
- ○ Para elaboração de Interface (TELA)
- ○ Para elaboração de Tabelas de Banco de Dados
- ○ Para elaboração de Regras de Negócio
- ○ Dúvidas relacionadas aos requisitos e entendimento do problema
- ○ Dúvidas relacionadas ao ambiente e infra-estrutura de desenvolvimento
- ○ Dúvidas relacionadas do projeto de software
- ○ Dúvidas relacionadas ao gerenciamento de projeto
- ○ Dúvidas de implementação
- ○ Nunca são consultados
- ○ Outros. Especifique_____

2.11. Quando ocorrem mudanças nos projetos, quais artefatos são alterados para refletir as alterações necessárias? *

Marque todas que se aplicam.

- o Artefatos de requisitos.
- o Artefatos de análise de projeto.
- o Artefato de implementação.
- o Não são realizadas alterações.

2.12. Nos projetos desenvolvidos na empresa, com qual frequência eles são representados por meios de modelos? *

	Sempre	Quase Sempre	Na maioria das vezes	Raramente	Nunca
Requisitos	○	○	○	○	○
Análise de Projeto	○	○	○	○	○
Construção	○	○	○	○	○

2.13. Na empresa, os códigos finais (códigos fonte, tabelas de banco de dados e interfaces gráficas) são gerados automaticamente ou semi-automaticamente a partir de quais artefatos? *

	Diagrama de Classe	Script de automatização	Diagrama de Comunicação	Diagrama de Sequência	Protótipo de Tela	Eclipse Modeling Framework (EMF)	Diagrama de Implantação	Não ocorre a geração	NPO
Geração de código fonte	☐	☐	☐	☐	☐	☐	☐	☐	☐
Geração de tabelas banco de dados	☐	☐	☐	☐	☐	☐	☐	☐	☐
Geração de Interface (Tela)	☐	☐	☐	☐	☐	☐	☐	☐	☐

3. MAPEAMENTO

3.1. Avalie as técnicas abaixo, considerando quanto cada uma favorece a interação do analista com o cliente para extração dos requisitos iniciais do sistema?

	NPO	1	2	3	4	5
VORD (Orientada a ponto de vista)	○	○	○	○	○	○
Observação	○	○	○	○	○	○
JAD (Joint Application Design)	○	○	○	○	○	○
BSP (Business System Planning)	○	○	○	○	○	○
Brainstorming	○	○	○	○	○	○
BPMN (Business Process Modeling Notation)	○	○	○	○	○	○
Questionários	○	○	○	○	○	○
Prototipação	○	○	○	○	○	○
Workshop	○	○	○	○	○	○
Storyboards	○	○	○	○	○	○
Cenário	○	○	○	○	○	○
Entrevista	○	○	○	○	○	○

(NPO- Não posso opinar) - Utilize-se das notas compreendidas entre 1(favorece pouco) e 5(favorece muito)

3.2. Avalie as formas de documentação abaixo, como cada uma favorece o entendimento e descrição dos requisitos? *

	Facilidade no Entendimento							Quando o desenvolvedor não participa da fase de requisitos e precisa compreendê-los para criar novos artefatos de etapas seguintes					
	NPO	1	2	3	4	5		NPO	1	2	3	4	5
História de Usuário	○	○	○	○	○	○		○	○	○	○	○	○
Método Formal	○	○	○	○	○	○		○	○	○	○	○	○
Diagrama de Atividade	○	○	○	○	○	○		○	○	○	○	○	○
Modelo de Caso de Uso	○	○	○	○	○	○		○	○	○	○	○	○
BPMN(Business Process Modeling Notation)	○	○	○	○	○	○		○	○	○	○	○	○
Diagrama de Máquina de Estado	○	○	○	○	○	○		○	○	○	○	○	○
Notação Kaos (Knowledge Acquisition in automated Specification)	○	○	○	○	○	○		○	○	○	○	○	○

(NPO- Não posso opinar) - Utilize-se das notas compreendidas entre 1(favorece pouco) e 5(favorece muito)

3.3. Dentre as representações da visão estrutural (estática), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento? *

	Adequada para o Entendimento							Adequadas para realizar a descrição					
	NPO	1	2	3	4	5		NPO	1	2	3	4	5
Linguagens de descrição de arquitetura (ADLs): linguagens textuais.	○	○	○	○	○	○		○	○	○	○	○	○
Diagramas de classes e objetos.	○	○	○	○	○	○		○	○	○	○	○	○
Diagramas de implantação.	○	○	○	○	○	○		○	○	○	○	○	○
Diagramas de entidade relacionamento.	○	○	○	○	○	○		○	○	○	○	○	○
Linguagens de descrição de interface (IDLs): linguagens de programação usadas para definir as interfaces.	○	○	○	○	○	○		○	○	○	○	○	○
Gráficos de estrutura.	○	○	○	○	○	○		○	○	○	○	○	○

(NPO- Não posso opinar) - Utilize-se das notas compreendidas entre1(favorece pouco) e 5(favorece muito)

3.4. Dentre as representações visão comportamental (dinâmica), avalie quanto cada uma favorece o entendimento e a descrição das informações necessárias para as fases posteriores do desenvolvimento? *

	Adequada para o Entendimento							Adequadas para realizar a descrição					
	NPO	1	2	3	4	5		NPO	1	2	3	4	5
Diagramas de atividade.	○	○	○	○	○	○		○	○	○	○	○	○
Diagramas de comunicação.	○	○	○	○	○	○		○	○	○	○	○	○
Fluxogramas.	○	○	○	○	○	○		○	○	○	○	○	○
Diagramas de sequência.	○	○	○	○	○	○		○	○	○	○	○	○
Diagramas de estado.	○	○	○	○	○	○		○	○	○	○	○	○
Linguagens de especificação formal	○	○	○	○	○	○		○	○	○	○	○	○
Pseudocódigo e linguagens de design de programa (PDLs)	○	○	○	○	○	○		○	○	○	○	○	○

(NPO- Não posso opinar) - Utilize-se das notas compreendidas entre 1(favorece pouco) e 5(favorece muito)

3.5. Avalie quanto a geração de código automático favorece os itens abaixo: *

	NPO	0	1	2	3	4	5
Padronização de código	○	○	○	○	○	○	○
Facilidade de entender o código	○	○	○	○	○	○	○
Produtividade	○	○	○	○	○	○	○
Manutenção do código	○	○	○	○	○	○	○

(NPO- Não posso opinar) - Utilize-se das notas compreendidas entre 0(favorece pouco) e 5(favorece muito)

APÊNDICE B- CONTIVE VIA E-MAIL

Assunto do e-mail: Convite para Colaboração em Pesquisa

Prezado(a) "Nome da Empresa",

O Grupo de Engenharia de Software e Gestão do Conhecimento da Universidade Estadual do Norte do Paraná - UENP, por meio dos pesquisadores José Maria Clementino Junior e o prof. Dr. André Menolli, está realizando uma pesquisa sobre o uso do desenvolvimento dirigido à modelos em empresas de desenvolvimento de software. Gostaríamos da sua participação nesta pesquisa, respondendo o questionário no link abaixo. O tempo médio de resposta é de 11 minutos.

Link para Pesquisa

Qualquer dúvida relativa a pesquisa pode entrar em contato conosco por meio do e-mail geskm@uenp.edu.br

Agradecemos antecipadamente sua participação.

APÊNDICE C- CONVITE VIA LINKEDIN

Saudações “Nome da pessoa”, tudo certo?

Sou integrante do Grupo de Pesquisa: Engenharia de Software e Gestão do Conhecimento da Universidade Estadual do Norte do Paraná - UENP, estamos desenvolvendo uma pesquisa relacionado ao desenvolvimento de software.

Gostaria muito de contar com sua colaboração para responder o questionário.

Desde já meu muito obrigado.

Link para Pesquisa