Aplicando Design Thinking em Engenharia de Software: um Mapeamento Sistemático

Anderson Felipe Barros de Souza, Bruna Moraes Ferreira, Tayana Conte

USES Research Group, Instituto de Computação, Universidade Federal do Amazonas, Av. Rodrigo Otávio, 6200, Coroado – CEP: 69077-000 – Manaus/Amazonas, Brasil
{anderson.souza, bmf, tayana}@icomp.ufam.edu.br

Abstract. Contexto: vários problemas na construção de software estão relacionados a falta de envolvimento do usuário no processo de desenvolvimento. Com isso, Design Thinking (DT) surge como uma metodologia para elicitação das reais necessidades do usuário produzindo serviços e produtos inovadores. **Objetivo:** investigar e compreender como DT é aplicado em Engenharia de Software. **Método**: foi realizado um mapeamento sistemático da literatura sobre o uso de Design Thinking em Engenharia de Software, extraindo informações de sua aplicação no processo de desenvolvimento. **Resultados:** o mapeamento sistemático apresenta 11 modelos de Design Thinking. Além disso, 10 ferramentas de software e 55 técnicas de DT são apresentadas. **Conclusão:** Design Thinking é uma metodologia que não segue necessariamente uma ordem entre as suas fases. Os modelos podem ser adaptados de acordo com o contexto do problema e suas técnicas auxiliam no processo de inovação do produto final.

Palavras-chave: Design Thinking, Mapeamento Sistemático, Processo de Inovação em Software, Técnicas de Design Thinking

1 Introdução

Um dos problemas que a Engenharia de Software tem tentado resolver, a partir do seu início, é como transformar um problema em possíveis soluções com uma orientação metodológica [36]. A falta ou pouco envolvimento dos usuários pode ser uma das causas desse problema. Segundo Kaur e Sengupta [23], o envolvimento dos usuários ao longo do desenvolvimento do software é um dos critérios que contribuem para o sucesso do projeto. Entender os usuários é essencial para projetar softwares que atendam às suas necessidades e expectativas [10]. Uma forma de manter o foco nos usuários durante o desenvolvimento do software é com a utilização da metodologia Design Thinking (DT) [29].

Design Thinking pode ser definido como uma metodologia utilizada por designers ao abordar problemas e pode ser aplicado em todas as áreas do conhecimento a fim de alcançar a inovação [37]. De acordo com Paula e Cormican [30], DT apresenta uma alternativa às abordagens típicas para resolução de problemas organizacionais, que consistem em várias etapas incluindo definição do problema, geração e teste de soluções. No contexto de Engenharia de Software, Design Thinking fornece uma metodo-

© Springer-Verlag Berlin Heidelberg 2011

logia para elicitação das necessidades do usuário e produz uma série de protótipos que normalmente convergem para soluções inovadoras [36].

Como forma de auxiliar a metodologia de Design Thinking, existem técnicas e ferramentas disponíveis. Chasanidou *et al*. [11] afirmam que um grande número de técnicas e ferramentas de design facilitam o processo de inovação de DT e selecionar os métodos corretos é importante especialmente nas fases iniciais. Saber quais são e como são aplicados estes métodos e ferramentas de DT em Engenharia de Software possibilita o conhecimento de novas alternativas para o processo de desenvolvimento e pode envolver o usuário de forma mais efetiva nas etapas de desenvolvimento de software.

O objetivo deste trabalho é apresentar os resultados de um mapeamento sistemático onde foram identificadas publicações que especificam modelos de Design Thinking em Engenharia de Software, bem como técnicas e ferramentas que auxiliam na aplicação de DT. Este artigo está organizado da seguinte forma: na Seção 2 é apresentado o referencial teórico sobre conceitos de Design Thinking. A Seção 3 apresenta o método de pesquisa utilizado. A Seção 4 descreve os resultados obtidos com o mapeamento. A Seção 5 discute os resultados obtidos. Finalmente, na Seção 6 são apresentados as conclusões e os trabalhos futuros.

2 Referencial Teórico

2.1 Design Thinking

Nas últimas duas décadas, Design Thinking amadureceu consideravelmente ganhando popularidade em vários campos, e é considerado um paradigma novo que lida com problemas em várias áreas como Tecnologia da Informação (TI), Educação, Negócios e Medicina [15]. Dunne e Martin [16] afirmam que o Design Thinking é a forma como os designers pensam e aplicam seus processos mentais para projetar objetos, serviços ou sistemas. DT combina abordagens e resolução de métodos utilizados por designers com pontos de vista e práticas de tecnologia e negócios [27].

As práticas de Design Thinking ajudam as organizações a resolverem problemas complexos, incentivando a inovação e apoiando o processo criativo [26]. Desta forma, DT é apresentada como uma metodologia adequada ao incentivo à inovação e ao crescimento econômico [26]. Por sua vez, a inovação resulta da elaboração de uma solução criativa que é desejável para o cliente e viável do ponto de vista econômico e tecnológico [37].

No contexto da Engenharia de Software, DT é importante, pois como a indústria moderna de software tornou-se altamente competitiva, há vários produtos do mesmo domínio de aplicações que competem entre si para atender aos usuários [7]. Para manter-se no mercado, o software precisa distinguir-se dos outros produtos similares e satisfazer seus clientes, proporcionando novidade e utilidade [6]. Como resultado, há necessidade de criar requisitos inovadores para equipar o software com uma vantagem competitiva [7].

2.2 Modelos de Design Thinking

Grande parte da literatura se concentra no desenvolvimento de modelos de DT para servir de guia para entendimento e resolução de problemas em diferentes contextos [30], por exemplo, contextos em que equipes de projeto são inexperientes. Modelos de Design Thinking geralmente são compostos por fases, algumas ajudam no entendimento do problema, outras na observação e empatia do usuário, entre outras características.

Paula e Cormican [30] realizaram um mapeamento sistemático da literatura sobre Design Thinking em estudos de design, que identificou modelos de DT considerando o seu foco, formato, diferenças e público-alvo. De acordo com este mapeamento, o modelo mais conhecido é o modelo proposto por Brown [8], que pode ser usado em diferentes contextos. Este modelo tem formato cíclico de três fases que se concentra na descoberta de novas oportunidades e solução de problemas [30]. Outro modelo que os autores destacam é o de Plattner *et al*. [31], que tem seis etapas interligadas e é bem usado em treinamento de equipes inexperientes. Além destes dois modelos, outros modelos também foram encontrados no mapeamento de Paula e Cormican [30]. Porém, foram identificados modelos de DT aplicados apenas em Design, diferentemente do mapeamento apresentado neste artigo, que buscou modelos de DT considerando somente a aplicação destes em Engenharia de Software.

3 Método de Pesquisa

A metodologia utilizada na execução deste mapeamento sistemático segue as orientações de Kitchenham e Charters [24]. A **Fig. 1** ilustra o método de pesquisa.

(1) Questão de Pesquisa: A questão de pesquisa foi formulada para que pudesse abranger todo o escopo de Design Thinking em Engenharia de Software. Neste contexto, a questão de pesquisa deste mapeamento sistemático é a seguinte: *"Como Design Thinking é aplicado em Engenharia de Software? "*.

A partir dessa questão de pesquisa foram definidas três subquestões:

- *Quais os modelos utilizados em Design Thinking?*
- *Quais as técnicas métodos utilizadas dentro dos modelos de Design Thinking?*
- *Quais as ferramentas existentes para apoiar a utilização das técnicas de Design Thinking?*

(2) Estratégia e Processo de Busca: As bibliotecas digitais utilizadas como fonte de busca foram Scopus[1] e da Engineering Village[2]. A Scopus é uma das bases que indexa os principais periódicos de Engenharia de Software [12]. A Engineering Village agrega informações de diversos bancos de dados bibliográficos em Ciência da Computação, abrangendo importantes periódicos e conferências da IEEE, ACM, Springer e Elsevier [35]. A string utilizada para busca nas bibliotecas foi a seguinte:

---

1 https://www.scopus.com/

2 http://www.engineeringvillage.com/

(“design thinking”) AND (“software engineering” OR “software development” OR “software industry” OR “systems development”)

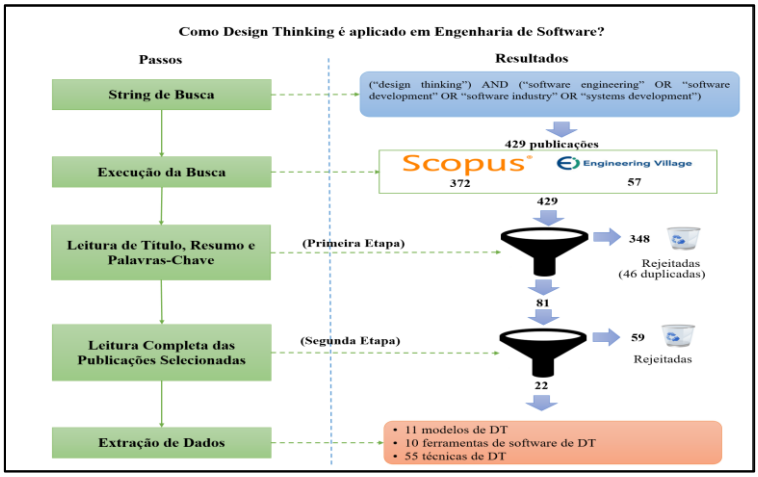

Fig. 1. Processo do Mapeamento Sistemático

Para avaliar a qualidade e abrangência da *string* de busca, foi realizada uma pesquisa exploratória na qual foram definidos 5 artigos de controle, que são apresentados no Relatório Técnico [33]. Após a execução da *string* de busca nas bibliotecas digitais, verificou-se que os artigos de controle estavam entre as publicações retornadas.

Para organizar as informações das publicações retornadas como resultado da busca foi utilizada a ferramenta Start [17], que auxilia no processo de revisão sistemática. Os resultados que a *string* de busca retornou durante a sua execução nas fontes de pesquisa foram agrupados nesta ferramenta para seleção das publicações de acordo com os critérios de inclusão e exclusão e para o processo de extração de dados.

(3) Critérios de Inclusão/Exclusão e Procedimentos de Seleção: Para a seleção das publicações foram definidos critérios de inclusão e exclusão. Estes critérios são apresentados na Tabela 1.

Tabela 1. Critérios de Seleção das publicações

Critérios de Inclusão
A publicação apresenta ferramentas computacionais, métodos, modelos ou abordagens baseadas em Design Thinking utilizados em Engenharia de Software.
A publicação apresenta uma discussão sobre a inclusão de Design Thinking no processo de desenvolvimento de software.
A publicação apresenta casos de uso de Design Thinking em organizações ou na academia utilizados no contexto de Engenharia de Software.
Critérios de Exclusão
Não serão selecionadas publicações que não atendam os critérios de inclusão.
Não serão selecionadas publicações duplicadas.
Não serão selecionadas publicações que não estão disponíveis para leitura.
Não serão selecionadas publicações que não estão nos idiomas português e inglês.

O procedimento adotado na seleção das publicações foi dividido em duas etapas. Na primeira etapa, foi realizada a leitura de título, resumo e palavras-chave de todas as publicações. Ao fim desta etapa, as publicações que não atendiam aos critérios de inclusão do mapeamento foram excluídas, considerando os critérios de exclusão. Se houvesse alguma dúvida sobre a relevância da publicação, a mesma era incluída para a próxima etapa.

Na segunda etapa, foi realizada a leitura completa das publicações potencialmente relevantes selecionadas na etapa anterior. Nesta seleção foram considerados os mesmos critérios de inclusão e exclusão. Ao final desta etapa, foi obtido um conjunto de publicações relevantes.

A busca realizada nas bibliotecas digitais retornou 429 publicações, 372 pela Scopus e 57 pela Engineering Village. Para aumentar a confiabilidade da classificação das publicações foi separada aleatoriamente uma amostra de 10% das publicações retornadas. Essa amostra foi classificada por dois pesquisadores individualmente para verificar o nível de concordância entre eles. Para medir a intensidade da concordância entre os dois pesquisadores, foi calculado o Cohen's Kappa [18]. O Kappa é uma medida de concordância interobservador e mede o grau de concordância além do que seria esperado tão somente pelo acaso [25]. O Kappa obtido para a classificação teve nível de concordância igual a 0.853. De acordo com a interpretação para os resultados do Kappa sugerido por Landis e Koch [25], o resultado obtido indica nível de concordância quase perfeita entre os pesquisadores.

Com a execução da primeira etapa 348 publicações foram rejeitadas, sendo 46 duplicadas. Considerando o número de publicações rejeitadas por biblioteca digital, tem-se pela Scopus 301, sendo 7 duplicadas, e 47 pela Engineering Village, sendo 39 duplicadas. Já considerando o número de publicações aceitas, foi obtido um total de 81 publicações para a segunda etapa, das quais 71 são pela Scopus e 10 pela Engineering Village.

Das 81 publicações, 22 foram aceitas, sendo 20 pela Scopus e 2 pela Engineering Village. O total de publicações rejeitadas é 59, das quais 51 são pela Scopus e 8 pela Engineering Village. A Tabela 2 resume o que foi realizado na primeira e na segunda etapa. A lista de todas as publicações extraídas está disponível no Relatório Técnico [33].

(4) Processo de Extração de Dados: O processo de extração de dados foi realizado com o auxílio de um formulário que contém itens relacionados aos dados que se deseja obter das publicações selecionadas. O formulário de extração de dados com todos os itens encontra-se no Relatório Técnico [33].

Tabela 2. Resultados da primeira e segunda etapa do mapeamento sistemático

Bibliotecas Digitais	**Publicações Retornadas**	**Publicações Selecionadas**	
		Primeira Etapa	**Segunda Etapa**
Scopus	372	71	20
Engineering Village	57	10	2
Total de Publicações Extraídas			22

4 Resultados

Ao verificar as publicações selecionadas, em relação ao número de publicações por ano, pode-se verificar que grande parte das publicações é recente, ou seja, 45% das publicações é de 2016. Isso significa que as pesquisas sobre Design Thinking em Engenharia de Software vêm crescendo desde 2011 (ano da publicação mais antiga identificada neste mapeamento sistemático). A **Fig. 2** apresenta o número de publicações identificadas por ano, de 2011 a 2016.

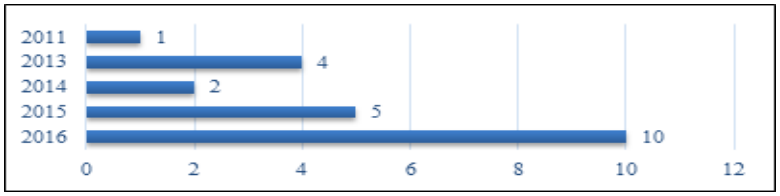

Fig. 2. Quantidade de publicações identificadas por ano

Considerando estas publicações, foram extraídas informações que respondem à questão de pesquisa através das subquestões definidas.

4.1 Quais os modelos utilizados em Design Thinking?

Durante a extração de dados foi possível identificar modelos de Design Thinking. No total, foram identificados 11 modelos de Design Thinking, que são apresentados na **Fig. 3**. De acordo com a figura pode-se observar que 2 modelos foram os mais citados, 5 e 4 vezes, respectivamente, enquanto que o restante foi citado apenas uma vez cada.

Entre os modelos mapeados, identificou-se que estes adotam fases para a sua execução. Por exemplo, o modelo apresentado por Sandino *et al*. [32] e apresenta sete fases. Este modelo é adaptado para aplicações em tempo real e apresenta como fases: definir, explorar, idealizar, prototipar, escolher, implementar e revisar. A **Fig. 4** mostra todos os modelos identificados com suas respectivas fases.

Também foram mapeados modelos que utilizam Design Thinking associado a outras tecnologias ou processos de desenvolvimento, como metodologia Ágil e Lean Startup. Estas também são metodologias alternativas às abordagens tradicionais para desenvolvimento de software. O primeiro modelo identificado é o modelo de Nordstrom, apresentado por De Paula e Araújo [14]. Este modelo inicialmente utilizava Design Thinking somente no início do modelo e depois vinham as fases dos métodos de Desenvolvimento Ágil e Lean Startup. Após modificações, De Paula e Araújo [14] propuseram um novo modelo Nordstrom, que expandiu o Design Thinking para todo o processo.

Já o DrivingBoard [28], é um componente de Design Thinking que compõe um framework chamado Speedplay. Além de Design Thinking este framework utiliza Design Participativo e Desenvolvimento Ágil. O DrivingBoard é um modelo de DT

que compreende as seguintes fases: abordar, desenvolver, apresentar e provocar, explorar, refletir e escapar. Por fim, o último modelo identificado é o modelo Converge [37], que combina Design Thinking, Lean Startup e Desenvolvimento Ágil. O Design Thinking utilizado no modelo Converge é o Stanford University's D-School.

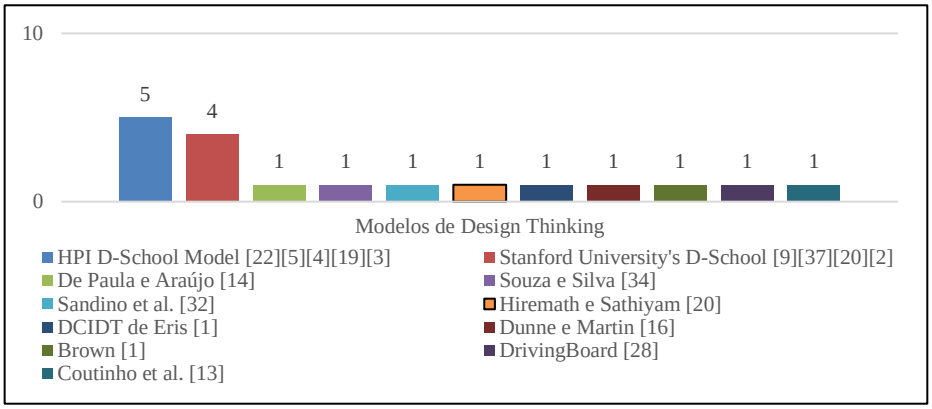

Fig. 3. Quantidade de citações para cada modelo de DT identificado

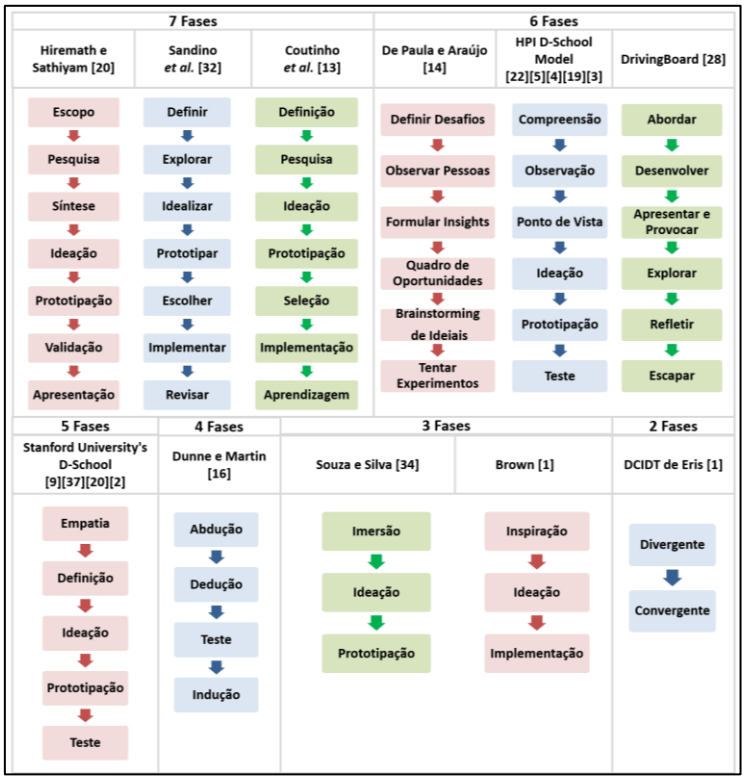

Fig. 4. Modelos de Design Thinking e suas fases.

4.2 Quais as técnicas/métodos utilizados dentro dos modelos de Design Thinking?

As técnicas de DT são fundamentais para sua execução. Este mapeamento identificou 55 técnicas entre as publicações. As técnicas podem ser usadas em uma ou mais fases do Design Thinking. Além disso, em uma única fase pode-se usar uma ou mais técnicas, dependendo do contexto e do problema.

As técnicas foram classificadas pela quantidade de citações nas publicações. Entre as técnicas mais citadas estão o Brainstorming, com 7 citações, e Persona, com 5 citações. Storyboard foi citada 4 vezes. As técnicas Customer Journey Map, Mapa Mental e Entrevista foram citadas 3 vezes cada uma. As técnicas Story Telling, Serviço de Blueprint e Cartões de Insight foram citadas 2 vezes cada uma como mostra a **Fig. 5**. As outras técnicas foram citadas apenas uma vez e são apresentadas na Tabela 3.

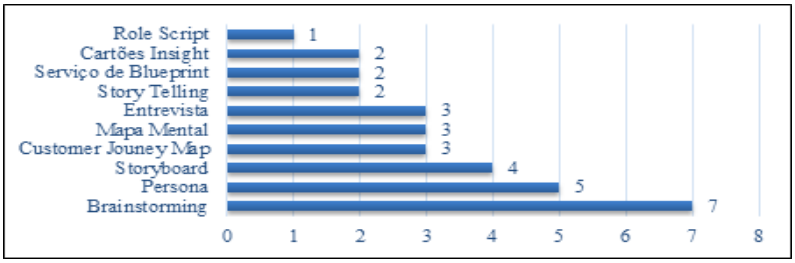

Fig. 5. Técnicas de Design Thinking mais citadas

Tabela 3. Técnicas de Design Thinking com apenas uma citação.

Nº	Técnica	Nº	Técnica
1	5 por que	24	Mapeamento Comportamental
2	Activity Analysis	25	Matriz de Motivação
3	Análise de Erro	26	Mockup
4	Arqueologia Comportamental	27	Moodboard
5	BodyStorming	28	Narração
6	Brainwalking Desencadeado	29	Character Profiles
7	Bussiness Model Innovation	30	Pesquisa Exploratória
8	Cartões de Questões	31	Poster
9	Casos de Uso	32	Prototipação Rápida
10	Diagrama de Afinidade	33	Prototipagem Brusca
11	Especificação	34	Protótipo de Experiência
12	Etnografica Rápida	35	Protótipo de Serviço
13	Evidencing	36	Questionário de Escala Likert
14	Fly On The Wall	37	Representação de Esboços
15	Grade de Análise de Tarefas	38	Role Play
16	Group Sketching	39	Role Script
17	Lego Serious Play	40	Serviço de Imagem

Nº	Técnica	Nº	Técnica
18	Mapa Conceitual	41	Mapeamento de Rede Social
19	Mapa de Atores	42	Stakeholder Map
20	Mapa de Empatia	43	Surveys e Questionários
21	Mapa de Oferta	44	Tomorrow Headlines
22	Mapa de Sistema	45	Touchpoints Matrix
23	Mapas Cognitivos	46	Try It Youself

4.3 Quais as ferramentas existentes para auxiliar a utilização das técnicas de Design Thinking?

Nas publicações selecionadas no mapeamento sistemático, foram citadas dez ferramentas, listadas na Tabela 4. Para cada ferramenta foi especificada as possíveis fases de Design Thinking, bem como a técnica em que a ferramenta auxilia. Somente as ferramentas NVivo e Balsamiq não especificaram a técnica que podem auxiliar. Por sua vez, Stakeholder Circle e Creately não especificaram a fase de DT em que podem ser utilizadas.

Tabela 4. Ferramentas de apoio à DT identificadas

Ferramenta	Fase de DT que pode ser usada	Técnica de DT
Rabiscapp[3] [37]	Prototipação	Representação de Esboços
Nvivo [34]	Imersão e Ideação	Não especificado
Balsamiq [34]	Prototipação	Não especificado
Smaply [11]	Empatia e Definição	Persona
Stakeholder Circle [11]	Não especificado	Stakeholder Map
Tachpoint Dashboard [11]	Empatia	Customer Journey Map
Creately [11]	Não especificado	Serviço de Blueprint
Strategyzer [11]	Ideação	Business Model Innovation
Axure RP [11]	Prototipação	Prototipação Rápida
Scenes [21]	Não especificado	Storyboard

5 Discussão dos Resultados

Com os resultados obtidos é possível fazer algumas observações em relação ao uso dos modelos de Design Thinking mapeados em Engenharia de Software e sobre a semelhança entre suas fases, bem como o uso das técnicas de DT. Ao observar os 3 modelos com 7 fases cada um, pode-se perceber que há uma semelhança entre algumas de suas fases. As fases de escopo [20], definir [32] e definição [13] se referem ao entendimento do problema que se quer resolver com o software a ser projetado, seja no projeto, seja no ambiente, seja em relação aos membros da equipe de desenvolvimento, definição de problema e público-alvo. Uma precisa compreensão do problema

[3] Ferramenta não está disponível

possibilita o desenvolvimento de soluções mais exatas [13]. Técnicas como entrevistas, personas, mapas de empatia e 5 porquês podem ser usadas nestas fases.

As fases de pesquisa [13][20] e explorar [32] reúnem informações acerca do que vai ser trabalhado, como identificação de potenciais usuários e suas necessidades, além de observação de soluções anteriores para o mesmo problema. Para isso são utilizadas técnicas como Social Networking Mapping, Surveys e Questionários e Etnografia Rápida. Em particular, o modelo de Hiremath e Sathyiam [20] apresenta uma fase a mais para a compreensão do problema, que é a fase de síntese. Nesta fase, as equipes de projeto trabalham no sentido de ver pontos de contato, ou seja, as interações dos usuários com a aplicação que será projetada, como Customer Journey Map e Mapas Mentais.

As fases de ideação [13][20] e idealizar [32] são fases de geração e seleção de ideias para gerar possíveis soluções. Motivações e necessidades do usuário final, identificadas anteriormente, são analisadas e a partir disso, ideias são geradas [13]. A técnica mais citada nestas fases é o brainstorming, mas outras técnicas podem ser usadas, como a Arqueologia Comportamental, que é uma técnica de observação em relação ao modo em como as pessoas usam e organizam o espaço e objetos onde acontece a atividade de projeto [32].

As próximas fases destes modelos de DT são semelhantes e se referem à prototipação. Nesta fase, as ideias geradas na fase de ideação são transformadas em protótipos. No caso da Engenharia de Software, seria o protótipo do sistema ou aplicação. Estes protótipos de ideias são apresentados às partes interessadas para revisão antes de serem apresentadas para o usuário final me ajuda. As técnicas que podem ser usadas nesta fase é a Prototipação Rápida e Representação de Esboços, por exemplo. Outra técnica que pode ser utilizada é o Storyboards.

As fases de seleção e implementação [13] são semelhantes às fases de escolher e implementar [32], que revisa as soluções propostas e implementa as que estão de acordo com o objetivo do projeto. Uma técnica que pode ser utilizada nesta fase é a Activity Analysis. No modelo de Hiremath e Sathyiam [20], fase semelhante no que diz respeito à escolha de soluções propostas é a fase de validação, que apresenta os protótipos aos usuários finais para desencadear discussões mais profundas e verificar se o software atende as necessidades do usuário e identificar possíveis melhorias para este software, ou seja, um teste de soluções. Uma particularidade no modelo de Coutinho *et al.* [13] é a fase de apresentação, que marca o encerramento do projeto e apresentação do modelo às partes interessadas. Além disso, este modelo é encerrado com a fase de aprendizagem, que auxilia os projetistas a melhorarem seu desempenho buscando feedback do cliente para saber se a solução atendeu às metas. Por sua vez o modelo de Sandino *et al.* [32] encerra com uma fase semelhante, que é a fase revisar. Nesta fase são identificadas melhorias e coleta de informações das pessoas que utilizam as aplicações, esta é uma fase de teste.

Hiremath e Sathyiam [20] afirmam que o seu modelo proposto foi adaptado para um projeto de 5 semanas e o modelo é baseado no modelo de Stanford D-School [9][37][20][2]. Devido à forte semelhança entre os modelos de 7 fases discutidos anteriormente percebeu-se que estes modelos podem ser uma evolução mais específica do modelo Stanford D-School ou modelo HPI D-School [22][5][4][19][3], conside-

rando um problema específico. O modelo Stanford D-School tem 5 fases e foi um dos mais citados entre os identificados no mapeamento. Suas fases são: empatia, definição, ideação, prototipação e teste. Já o modelo HPI D-School tem 6 fases: compreensão, observação, ponto de vista, ideação, prototipação e teste.

Entre os modelos de 6 fases identificados, o HPI D-School foi o mais citado no mapeamento sistemático. A fase compreender está relacionada ao entendimento do problema. Observação é relação a observação dos usuários para obter empatia deles. Ponto de vista é a síntese de insights que é estabelecida como uma base de conhecimento comum para as fases subsequentes. As próximas fases já foram discutidas.

Os outros modelos de 6 fases, DrivingBoard [28] e de De Paula e Araújo [14], são modelos mais específicos para um problema. O DrivingBoard segue uma linha de desenvolvimento totalmente adaptada para resolução de um problema social e não se sabe a partir de qual modelo de DT ele é derivado. Além disso, o DrivingBoard está associado ao Design Participativo e Metodologia Ágil, bastante utilizada no contexto de desenvolvimento de software. Já o modelo de De Paula e Araújo é semelhante ao HPI D-School, podendo ser considerada uma variação do HPI D-School adaptado.

O modelo de Dunne e Martin [16] apresenta 4 fases: abdução, dedução, teste e indução. Neste modelo, a fase de abdução se concentra na geração de ideias. Durante a fase de dedução, essas ideias são analisadas para prever prováveis consequências. Todas as previsões então são testadas e os resultados válidos são generalizados durante a fase de indução. Mais simples ainda, são os modelos de Brown [1] e em Souza e Silva [34], que apresentam 3 fases e são bastante semelhantes. A diferença é que o nome que Brown dá para a primeira fase de seu modelo é inspiração e Souza e Silva referem-se a esta fase como imersão, mas ambas estão relacionadas com a exploração do contexto do problema, levantamento, análise e síntese de dados. Por fim, o modelo de 2 fases, o DCIDT (Divergent-Convergent Inquiry Design Thinking), diverge sobre os vários conceitos do problema e depois converge as decisões criadas até chegar a uma decisão final.

Não foram encontrados no mapeamento evidências experimentais relacionadas aos modelos de Brown, Dunne e Martin e DCIDT em Engenharia de Software. Em relação aos outros modelos de Design Thinking discutidos e apresentados aqui, pode-se perceber que não existe um modelo absoluto que é utilizado para todos os tipos de problemas, ou seja, aquele modelo usado para todo tipo de problema. Notou-se também que é possível utilizar várias técnicas de DT combinadas em uma mesma fase de um modelo de DT e utilizar as mesmas técnicas em várias fases. Logo, um conjunto de técnicas é definido.

6 Conclusão

Este trabalho apresentou um mapeamento sistemático que teve como objetivo investigar a aplicação de Design Thinking em Engenharia de Software. Um conjunto de 429 publicações foi analisado, onde 22 foram selecionadas como publicações relevantes e para o mapeamento. Foram apresentados 11 modelos de Design Thinking e a quanti-

dade de citações de cada modelo, porém não é possível afirmar que os modelos mais citados são os mais usados em Engenharia de Software.

Além disso, foram listadas técnicas de Design Thinking, bem como o número de citações de cada uma. Foi possível observar que as técnicas podem ser usadas em conjunto em uma mesma fase. Mostrou-se também que ainda não há um grande número de ferramentas de software disponíveis que possam auxiliar o processo de DT, são apresentas apenas 10 ferramentas. Como trabalhos futuros, pretende-se realizar um estudo para verificar qual o melhor conjunto de técnicas para uma determinada fase de sum modelo de Design Thinking específico. Além disso, pretende-se realizar um estudo comparativo entre algumas ferramentas que apoiam o uso das técnicas de Design Thinking.

Referências

1. Adikari, S., Mcdonald, C., Campbell, J.: Reframed contexts: design thinking for agile user experience design. In: International Conference of Design, User Experience, and Usability. Springer Berlin Heidelberg. pp. 3-12 (2013).
2. Araújo, R., Anjos, E., Silva, D. R.: Trends in the Use of Design Thinking for Embedded Systems. In: ICCSA (Short Papers/poster papers/PhD student showcase works). pp. 82-86 (2015).
3. Berger, A.: Design thinking for search user interface design. Proceedings of euro-HCIR2011, pp. 1-4 (2011).
4. Beyhl, T., Berg, G., Giese, H.: Traceability recovery for innovation processes. In: Proceedings of the 8th International Symposium on Software and Systems Traceability. IEEE Press. pp. 22-28 (2015).
5. Beyhl, T., Berg, G., Giese, H.: Why innovation processes need to support traceability. In: Traceability in Emerging Forms of Software Engineering (TEFSE), 2013 International Workshop on. IEEE. pp. 1-4 (2013).
6. Bhowmik, T., Niu, N., Mahmoud, A., Savolainen, J.: Automated support for combinational creativity in requirements engineering. In: Requirements Engineering Conference (RE), 2014 IEEE 22nd International. IEEE. pp. 243-252 (2014).
7. Bhowmik, T., Niu, N., Savolainen, J., Mahmoud, A. Leveraging topic mode-ling and part-of-speech tagging to support combinational creativity in requi-rements engineering. In: Requirements Engineering, v. 20 n. 3, pp. 253-280 (2015)
8. Brown, B.: Design thinking. Harvard Business Review, v. 86, no. 6, pp. 84-92 (2008).
9. Carroll, N., Richardson, I.: Aligning healthcare innovation and software requirements through design thinking. In: Software Engineering in Healthcare Systems (SEHS), IEEE/ACM International Workshop on. IEEE. pp. 1-7 (2016).
10. Castro, J. W., Acuña, S. T., Juristo, N.: Enriching requirements analysis with the personas technique, In: International Workshop on: Interplay between Usability Evaluation and Software Development (I-USED 2008), pp. 13-18 (2018).
11. Chasanidou, D., Gasparini, A. A., Lee, E.: Design thinking methods and tools for innovation. In International Conference of Design, User Experience, and Usability. Springer International Publishing. pp. 12-23 (2015).
12. Costa, S. L., Pereira, V.: Uma Revisao Sistemática sobre as Ferramentas de Apoio do Método Z e da Notaçao Z. In: ESELAW'13: Proceedings of the 10th Experimental Software Engineering Latin American Workshop. pp. 34-47 (2013).

13. Coutinho, E. F., Gomes, G. A. M., Leite, A. J. M. Applying design thinking in disciplines of systems development. In: Telematics and Information Systems (EATIS), 8th Euro American Conference on. IEEE. pp. 1-8 (2016).
14. De Paula, D. F. O., Araújo, C. C.: Pet Empires: Combining Design Thinking, Lean Startup and Agile to Learn from Failure and Develop a Successful Game in an Undergraduate Environment. In: International Conference on Human-Computer Interaction. Springer International Publishing. pp. 30-34 (2016).
15. Dorst, K.: The nature of design thinking. In: Proceedings of the 8th Design Thinking research Symposium. pp. 19-20 (2010).
16. Dunne, D., Martin, R.: Design Thinking and How It Will Change Management Education: An Interview and Academy of Management Learning & Education, v. 5, n. 4, pp. 512-523 (2006).
17. Fabbri, S., Silva, C., Hernandes, E., Ocataviano F., Di Thommazo, A., Belgamo, A.: Improvements in the StArt tool to better support the systematic review process. In: the 20th International Conference, Limerick. Proceed-ings of the 20th International Conference on Evaluation and Assessment in Software Engineering - EASE '16. pp. 21:1-21:5 (2016).
18. Fleiss, J. L. Statistical Methods for Rates and Proportions, Second ed., John Wiley & Sons, New York (1981).
19. Gurusamy, K., Srinivasaraghavan, N., Adikari, S.: An Integrated Framework for Design Thinking and Agile Methods for Digital Transformation. In: International Conference of Design, User Experience, and Usability. Springer International Publishing. pp. 34-42 (2016).
20. Hiremath, M., Sathiyam, V.: Fast train to DT: a practical guide to coach design thinking in software industry. In: IFIP Conference on Human-Computer Interaction. Springer Berlin Heidelberg. pp. 780-787 (2013).
21. Jensen, M. B., Lozano, F., Steinert, M.: The Origins of Design Thinking and the Relevance in Software Innovations. In: Product-Focused Software Process Improvement: 17th International Conference, PROFES 2016, Trondheim, Norway, November 22-24, 2016, Proceedings 17. Springer International Publishing. pp. 675-678 (2016).
22. Kabiawu, O., Van Belle, J-P., Oshin, M. A.: Designing a Knowledge Resource to Address Bounded Rationality and Satisficing for ICT Decisions in Small Organizations.In: The Electronic Journal of Information Systems in Developing Countries, v. 73, n. 6, pp. 1-18 (2016).
23. Kaur, R., Sengupta, J.: Software Process Models and Analysis on Failure of Software Development Projects. International Journal of Scientific & Engineering Research, v. 2, n. 2, pp. 1-4 (2011).
24. Kitchenham, B., Charters, S.: Guidelines for performing systematic literature reviews in software engineering. In EBSE Technical Report EBSE-2007- 01, Software Engineering Group Department of Computer Science Keele University (2007).
25. Landis, J. R., Koch, G. G.: “The measurement of observer agreement for categorical data”. Biometrics; 33, pp. 159-174 (1977).
26. Liedtka, J.: Learning to use design thinking tools for successful innovation. In: Strategy & Leadership, v. 39, n. 5, pp. 13-19 (2011).
27. Matz, A., Germanakos, P.: Increasing the Quality of Use Case Definition Through a Design Thinking Collaborative Method and an Alternative Hybrid Documentation Style. In: International Conference on Learning and Collaboration Technologies. Springer International Publishing. pp. 48-59 (2016).
28. Newman, P., Ferrario, M. A., Simm, W., Forshaw, S., Friday, A., Whittle, J.: The role of design thinking and physical prototyping in social software engineering. In: Proceedings of

the 37th International Conference on Software Engineering-Volume 2. IEEE Press. pp. 487-496 (2015).
29. Ostrowski, S., Rolczyński, R., Pniewska, J., Garnik, I.:"User-friendly E-learning Platform: a Case Study of a Design Thinking Approach Use". In: Proceedings of the Mulitimedia, Interaction, Design and Innnovation. ACM. pp. 19-26 (2015).
30. Paula, D., K. Cormican.: Understanding Design Thinking in Design Studies (2006-2015): A Systematic Mapping Study. DS 84: Proceedings of the DESIGN 2016 14th International Design Conference. pp. 1-10 (2016).
31. Plattner, H., Meinel, C., Leifer, L. (Ed.).: Design thinking: understand–improve–apply. Springer Science & Business Media (2010).
32. Sandino, D., Matey, L. M., Vélez, G.: Design thinking methodology for the design of interactive real-time applications. In: International Conference of Design, User Experience, and Usability. Springer Berlin Heidelberg. pp. 583-592 (2013).
33. Souza, A. F. B., Ferreira, B. M., Conte, T. 2017. Mapeamento Sistemático sobre a aplicação de Design Thinking em Engenharia de Software. In USES Technical Report USES RT-USES-2017-0001. Disponível em: http://uses.icomp.ufam.edu.br/wp-content/uploads/2017/04/TR-USES-2017-0001.pdf.
34. Souza, C. L. C., Silva, C.: Uso do Design Thinking Na Elicitação De Requisitos De Ambientes Virtuais De Aprendizagem Móvel. 17th Workshop de Engenharia de Requisitos – WER. XVII Congresso Ibero Americano de Engenharia de Software. Pucón, Chile. pp. 1-14 (2014).
35. Trindade, C. C., Moraes, A. K., Meira, S. R. L.: Comunicação em equipes distribuídas de desenvolvimento de software: Revisão sistemática. In: ESELAW'08: Proceedings of the 5th Experimental Software Engineering Latin American Workshop. pp. 1-10 (2008).
36. Vetterli, C., Brenner, W., Uebernickel, F., Petrie, C.: From palaces to yurts: Why requirements engineering needs design thinking. IEEE Internet Computing, v. 17, n. 2, pp. 91-94 (2013).
37. Ximenes, B. H., Alves, I. N., Araújo, C. C.: Software Project Management Combining Agile, Lean Startup and Design Thinking. In: International Conference of Design, User Experience, and Usability. Springer International Publishing. pp. 356-367 (2015).