
**UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO**

**CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

# GUIA PARA SUPORTE À INSERÇÃO DE REQUISITOS DE SEGURANÇA EM PROJETOS ÁGEIS

DAYANNE CRISTINA DE ARAUJO BARBOSA

RECIFE

2015

# GUIA PARA SUPORTE À INSERÇÃO DE REQUISITOS DE SEGURANÇA EM PROJETOS ÁGEIS

Projeto apresentado como requisito para a nota da disciplina: Trabalho de conclusão de curso, ministrado pela Prof.ª Francielle Silva Santos no curso de Ciência da Computação na Universidade Federal Rural de Pernambuco - UFRPE

**Orientadora**: Prof.ª MSc. Suzana Cândido de Barros Sampaio

DAYANNE CRISTINA DE ARAUJO BARBOSA

# GUIA PARA SUPORTE À INSERÇÃO DE REQUISITOS DE SEGURANÇA EM PROJETOS ÁGEIS

Projeto apresentado como requisito para a nota da disciplina: Trabalho de conclusão de curso, ministrado pela pela Prof.ª Francielle Silva Santos no curso de Ciência da Computação na Universidade Federal Rural de Pernambuco - UFRPE

6 de fevereiro 2015.

**MINISTÉRIO DA EDUCAÇÃO E DO DESPORTO**
**UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO (UFRPE)**
**BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO**

http://www.bcc.ufrpe.br

## FICHA DE APROVAÇÃO DO TRABALHO DE CONCLUSÃO DE CURSO

Trabalho defendido por Dayanne Cristina de Araujo Barbosa como requisito para conclusão do curso de Bacharelado em Ciência da Computação da Universidade Federal Rural de Pernambuco, intituladoUm Guia para Suporte à Inserção de Requisitos de Segurança em Projetos Ágeis, orientado pelo Prof.Suzana Cândido de Barros SampaioSuzana Cândido de Barros Sampaio e aprovado pela seguinte banca examinadora:

Suzana Cândido de Barros Sampaio
DEINFO-UFRPE

Marcelo Luiz Monteiro Marinho
DEINFO-UFRPEDEINFO-UFRPE

Fernando Antônio Aires Lins
DEINFO-UFRPE

## AGRADECIMENTOS

À minha família que me ofereceu estrutura para correr atrás das minhas realizações. Especialmente à minha mãe, Cleide Araujo, por sua força, carinho e dedicação. À minha tia Iris, por ser uma segunda mãe. Ao meu pai, Djalma Araujo, que sempre me mandou estudar; À minha irmã, Ana Beatriz, pelos momentos de descontração. Aos meus tios, Otávio Nunes e Maria José de Araujo, meus agradecimentos pelo imenso apoio educacional. Minha família é a minha maior fonte de motivação.

Ao meu namorado, Húgaro Bernardino, pela paciência e amor sempre.

A Apolo e a Luke, pelos momentos de reflexão.

A Deus que, certamente, participou disso tudo me dando força, saúde e sabedoria.

Aos professores e companheiros da UFRPE - destaque B-hash - pela troca de conhecimento e momentos vivenciados; indubitavelmente foram essenciais para a realização deste trabalho. Em especial, ao companheiro Iago Moraes, pelas extraordinárias tragédias que enfrentamos e por termos feito graça de quase tudo. A Jamerson Lima e Thais Bione que foram companheiros de trabalho de conclusão de curso. A Alex Justino pelas interrupções deste trabalho para momentos criativos.

À minha orientadora, Suzana Sampaio, pela dedicação, puxões de orelha, disponibilidade, por acreditar e motivar meu trabalho, além de me ensinar a ser ágil para a vida.

A Fernando Aires por ter me iniciado cientificamente, dando suporte e motivação aos meus primeiros passos como pesquisadora;

Aos companheiros do CESAR que apoiaram e incentivaram minha carreira estudantil e profissional, contribuindo diretamente e indiretamente para a realização deste trabalho;

Ao café e à arte de viver da fé.

## RESUMO

As metodologias ágeis oferecem uma série de benefícios aos projetos. Estudos apontam que fatores de sucesso estão associados ao uso de tais metodologias. Dessa forma, a agilidade está sendo amplamente adotada por uma quantidade significativa de empresas. Em adição, Metodologias Ágeis são uma excelente alternativa para projetos que necessitam de adaptação a mudanças e inserção de requisitos durante a execução do projeto. Nesse sentido, projetos sob a perspectiva ágil podem funcionar com uma visão parcial ou incompleta do que será desenvolvido durante toda a execução do projeto. Isso dificulta a realização das práticas de Engenharia de Segurança, pois muitas delas foram propostas para Frameworks tradicionais. Em adição, segurança é extremamente importante para garantir o funcionamento de aplicações que utilizam dados sigilosos e que não devem ser acessados por usuários não autorizados. Assim, este trabalho tem como objetivo ampliar a qualidade dos produtos desenvolvidos, dando suporte e destaque à integração de requisitos de segurança, de forma simplificada, através de um guia para integração com as Metodologias Ágeis de desenvolvimento, a fim de ampliar a qualidade do produto desenvolvido. Isso pode dar suporte e destaque à integração dos requisitos não funcionais de segurança de forma simplificada através de um guia. O guia foi desenvolvido através de uma pesquisa exploratória, onde foi realizada uma exaustiva pesquisa no Google Scholar, pesquisando-se termos como: "*Extreme programming security practices", "XP security", "security agile", "security scrum", "extreme programming secure", "cost agile security", "xp security engineering*", entre outras combinações e variações de idioma. O conteúdo desta pesquisa foi submetido à análise e estudo comparativo para descobrir o estado da arte e fundamentar a construção para o guia. Ainda em estado de conclusão do levantamento dos dados do guia, foram realizadas algumas entrevistas preliminares com especialistas para discutir e orientar a construção do guia, bem como priorizar as práticas identificadas na pesquisa exploratória. Assim, foi possível desenvolver um guia para prover a integração entre métodos ágeis e segurança. Este guia foi avaliado através de novas entrevistas com especialistas. Dessa forma, este trabalho também apresenta discussões sobre as características positivas e negativas das sugestões do guia. Além da avaliação da utilidade e facilidade do guia em geral e de cada recomendação em particular.



Palavras-chave: Metodologias ágeis, Segurança, Integração segurança e agilidade, práticas ágeis, práticas de segurança.

## ABSTRACT

Agile methodologies offer many benefits to projects. Studies have shown that factors of success are associated with the use of such methodologies. Thus, the agility is being widely adopted by a significant number of companies. In addition, Agile methodologies are the best alternative for projects that need to adapt themselves to changes and insert new requirements during the project's implementation phase. In this sense, projects under the agile approach can work with a partial or incomplete view of what will be developed throughout its execution. This poses a challenge to the implementation of Security Engineering practices, as many of them have been proposed for traditional frameworks. In addition, security is extremely important to ensure that operations that deal with sensitive data will not be accessed by unauthorized users. Thus, this work aims to create a complement to Agile Methodologies development in order to increase the quality of their product, supporting and highlighting the integration of security non-functional requirements in a simplified way through a guide. The guide was developed through an exploratory research in which an exhaustive search in Google Scholar by terms like "Extreme programming security practices", "xp security", "security agile", "security scrum", "extreme programming secure", "cost agile security", "xp security engineering" amongst other combinations and language variations were applied. The contents of this research were subjected to analysis and to a comparative study to find out the state of the art and support for the construction of the Guide. Yet in a state of completion of the data collection, some preliminary interviews were done with experts to discuss and adjust the construction of the Guide and prioritize the practices identified in the exploratory research. Thus, it was possible to develop a guide to provide integration between agile methods and security. This guide has been valued through another set of interviews with experts. Therefore, this work also presents a discussion on the positive and negative characteristics of the guide's suggestions, besides the evaluation of the guide's and recommendations' general usefulness and ease.



Keywords: Agile methodologies, Security, Integration safety and agility, agile practices, security practices.

## LISTA DE FIGURAS

## LISTA DE QUADROS

## LISTA DE TABELAS

## LISTA DE ABREVIATURAS, SIGLAS E SÍMBOLOS

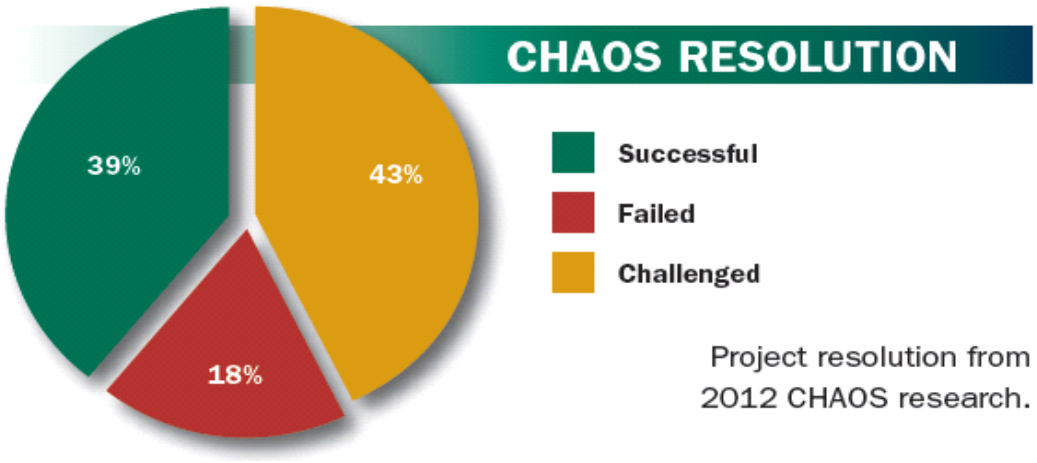
| | |
|---|---|
| CLASP | *Comprehensive, Lightweight Application Security Process* |
| FDD | *Feature Driven Development* |
| FISA-XP | *Framework Integrating Security Activities with Extreme Programming.* |
| OWASP | *Open Web Application Security Project* |
| SDL | *Security Development Lifecycle* |
| SQL | *Structured Query Language* |
| TISA-XP | *Tool Integrating Security Activities with Extreme Programming* |
| XP | *Extreme Programming* |

## SUMÁRIO

# 1. INTRODUÇÃO

A competitividade do mercado normalmente pressiona empresas a lançarem produtos rapidamente para os disponibilizarem antes de seus concorrentes ou em momentos estratégicos. Dessa forma, muitas vezes os projetos de *software* acabam sendo comercializados prematuramente, sem os devidos testes e avaliações (CORREIA; SOUSA, 2010). O relatório do *Chaos report* (STANDISH GROUP, 2013) mostra que a taxa de sucesso dos projetos em 2012 foi de 39%, como apresentado na Figura 1, ou seja, mais da metade dos projetos foram entregues de forma comprometida ou foram cancelados. Esse mesmo estudo aponta dez fatores de sucesso em projetos que estão relacionados às metodologias ágeis. Dentre esses fatores: o envolvimento do usuário, processos ágeis, claros objetivos de negócios e otimização, entre outros.

Figura 1 – *Chaos Resolution*: sobre sucesso em projetos




Fonte: Standish Group, 2013.

Alguns desses fatores de sucesso são princípios ou ainda valores descritos no Manifesto Ágil, publicado em 2001 (BECK et al., 2011). Desde então, organizações de desenvolvimento de *software* têm aderido a Metodologias Ágeis objetivando fazer parte desse contingente de sucesso. Pesquisas reforçam essa premissa (VERSION ONE, 2013; WILLIAMS, 2010), mostrando que empresas têm adotado Metodologias Ágeis como forma de gerir projetos de desenvolvimento de *software*. Assim, conquistando benefícios significativos, apresentando equipes mais produtivas, menos estressadas e clientes mais satisfeitos com os produtos entregues (HIGHSMITH, 2011). Em adição, empresas ágeis

conseguem ser 37% mais lucrativas e aumentar a receita 30% mais rapidamente comparadas às empresas não ágeis (TONIN et al, 2012).

Segundo pesquisa realizada em 2012 pela *Version One* (2013), o desenvolvimento ágil de *software* já é utilizado por 84% das empresas mundiais, sendo o principal modelo de produção de *software* em 59% destas companhias. Essa representatividade torna os frameworks ágeis referências para o mercado de software e motiva pesquisas em diferentes níveis. A pesquisa mostra ainda que alguns dos principais motivos de resistência à utilização de frameworks ágeis são a dificuldade de integrar pessoas ao modelo e à queda na qualidade do produto.

Um dos grandes desafios dos processos ágeis é agregar valor não funcional aos projetos sem comprometer outros aspectos, tais como tempo e custo. A natureza mais empírica e incremental dos processos ágeis de desenvolvimento gera uma carga de questionamentos que devem ser respondidos de maneira rápida e consistente a cada momento do projeto, relacionados aos aspectos não funcionais. Tais questões demandam de técnicas e ferramentas metodológicas adequadas para fornecer estas respostas (SOARES, 2006).

As Metodologias Ágeis lidam melhor com mudanças de requisitos do que as metodologias prescritivas. Dessa maneira, projetos desenvolvidos sob as Metodologias Ágeis trabalham com uma visão incompleta ou parcial do que deverá ser feito durante toda a execução do projeto. Isso dificulta a realização das práticas da Engenharia de Segurança em projetos ágeis para inserção de requisitos de segurança (BACA; CARLSSON, 2011). Essa dificuldade de integração entre as práticas de segurança e as Metodologias Ágeis acontece porque muitas das práticas de segurança não foram criadas sob a perspectiva das Metodologias Ágeis (BACA; CARLSSON, 2011; BOSTRÔM et al., 2006; KERAMATI; MIRIAN-HOSSEINABADI, 2008; WÄYRYNEN; BODÉN; BOSTRÖM, 2005).

Em adição, o uso da *Internet* e outras redes são extremamente importantes para sistemas distribuídos e usuários em geral. Entretanto, o aumento de riscos à segurança também estão associados ao uso dessas tecnologias (CORREIA; SOUSA, 2010). Nesse contexto, segurança é definida como o conjunto de medidas para proteger o sistema (IETF, 2015). São as técnicas de segurança que permitem, por exemplo, dados sigilosos como números de cartões de créditos e senhas trafeguem através da *Internet* de forma segura.

Neste cenário de crescente demanda por métodos ágeis e carência de engenharia de segurança, foi que algumas organizações e pesquisadores iniciaram estudos para união das duas áreas (ANDRADE; QUEIROZ; QUEIROZ, 2011; AYALEW; KIDANE, 2012;

AZHAM; GHANI; ITHNIN, 2011; BACA; CARLSSON, 2011; BARTSCH, 2011; BEZNOSOV, 2003; BEZNOSOV; KRUCHTEN, 2005; BOSTRÖM, 2006; GE et al., 2006; GHANI; YASIN, 2013; KERAMATI; MIRIAN-HOSSEINABADI, 2008; PEETERS, 2005; SONIA; SINGHAL; BANATI, 2014; STEVEN; PETERSON, 2007; WILLIAMS, 2010).

Contribuindo para essa linha de pesquisa, este trabalho apresenta uma revisão bibliográfica sobre as duas áreas e um guia para suporte à inserção de requisitos de segurança em projetos ágeis.

## 1.1 JUSTIFICATIVA

Com o advento do crescimento na adoção dos métodos ágeis, alguns requisitos não funcionais, como segurança, acabam negligenciados por não ser inserida no processo de desenvolvimento, da mesma forma que requisitos funcionais (KERAMATI, 2008). Nesse sentido, as práticas de Engenharia de Segurança precisam de atenção especial para viabilizar a construção de *software* seguro. Este trabalho apresenta um guia para suporte à inserção de requisitos e práticas de Engenharia de Segurança em projetos ágeis.

## 1.2 ELABORAÇÃO DOS OBJETIVOS

### 1.2.1 Geral

Este trabalho teve como objetivo principal propor e desenvolver um guia para as Metodologias Ágeis de desenvolvimento, de modo a ampliar a qualidade do produto desenvolvido, dando suporte e destaque à integração dos requisitos não funcionais de segurança de forma simplificada através de um guia.

### 1.2.2 Específicos

Para que fosse possível atingir o objetivo principal, os seguintes objetivos específicos foram realizados:

- Pesquisar, investigar referências de trabalhos ou relatos de experiências focados no desenvolvimento ágil relacionados à segurança.
- Identificar o que pode ser utilizado junto ao desenvolvimento ágil de *software* para melhor suporte à segurança.

- Propor um guia para inserir práticas de segurança no desenvolvimento de *software* ágil.

## 1.3 ESTRUTURA DO TRABALHO

Este trabalho está organizado em seções que tratam de aspectos cujo enlace converge em torno da inserção de requisitos de segurança em projetos ágeis. Assim, esta parte introdutória, além de contextualizar o tema, ressuscita os objetivos que nortearam a pesquisa, trazendo também uma sucinta justificativa, realçando a relevância da escolha deste tema.

A segunda parte apresenta a Metodologia, pormenorizando vários elementos essenciais adotados no estudo que resultou este texto. Nessa seção são brevemente apresentadas as etapas percorridas para a execução deste trabalho.

Uma terceira seção comenta sobre dados relacionados às Metodologias Ágeis e à segurança. Também é feita a elucidação sobre a união entre Metodologias Ágeis e segurança e fatores necessário para a melhor compreensão das recomendações do guia. Nesse capítulo, também são ressaltados os principais aspectos encontrados nos trabalhos analisados para fundamentar este texto.

Aspectos diretamente ligados à estrutura e organização do guia para inserção de práticas de segurança em metodologias ágeis como descrições das recomendações, benefícios e exemplos de trabalhos relacionados que fundamentaram o guia aparecem na quarta seção.

A quinta seção do texto apresenta as opiniões e perspectivas de especialistas em agilidade e segurança. Nesse sentido, esses dados são utilizados para a avaliação do guia e suas recomendações.

As conclusões no final do trabalho ainda realçam fatores relevantes ao tema e contribuições, além de sugerir contribuições futuras através de eventuais trabalhos que possam ser desenvolvidos a partir deste trabalho.

## 2. DEFINIÇÃO DA METODOLOGIA

A natureza dessa pesquisa melhor se adequou à abordagem exploratória, pois tal estilo forneceu maior compreensão do objeto de estudo através do aprimoramento de ideias e facilitou a construção de hipóteses (GIL, 2006). Esse tipo de pesquisa tende utilizar casos de uso e análises qualitativas (WAZLAWICK, 2008).

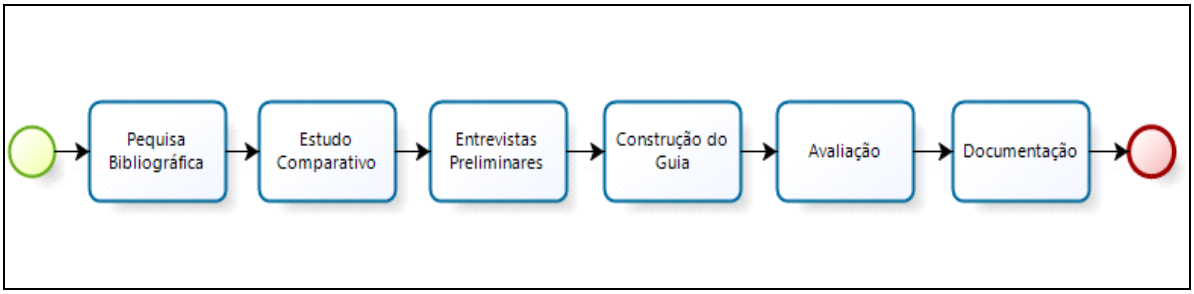
Durante a pesquisa foi realizada uma busca *ad hoc* por trabalhos recentes (de 2010 até 2014) sobre o tema deste trabalho. O *Google* (2015) e o *Google Scholar* (2015) foram utilizados como ferramentas, através deles foi possível acessar trabalhos disponíveis em diversos sites. Para a pesquisa, as seguintes palavras-chave foram utilizadas: *extreme programming, security practices, xp security, security agile, security scrum, extreme programming secure, cost agile security, xp security engineering*, entre outras combinações e variações de idioma. Alguns trabalhos relacionados ao tema foram encontrados, porém, não foi possível acessá-los.

Assim foi possível compreender, analisar e comparar os trabalhos através de leitura dos achados. Durante a análise, outros trabalhos relevantes mais antigos foram encontrados sendo referenciados. Esses trabalhos mais antigos foram selecionados para estudados assim como os trabalhos mais atuais. Feito isso, algumas entrevistas foram feitas previamente com especialistas para maior precisão sobre a relevância prática dos trabalhos analisados.

Baseando-se nas informações alcançadas através da execução das fases anteriores, o guia foi o principal resultado desta pesquisa oferecendo abordagem para a integração entre agilidade e segurança. A avaliação do guia proposto ocorreu através de uma pesquisa qualitativa, utilizando-se técnicas como entrevistas.

Na Figura 2, é possível ver um fluxo, demonstrando as etapas percorridas para a construção do guia, usando o contexto do cronograma do projeto.

Figura 2 – Etapas percorridas para a construção do guia




Fonte: Arquivo da autora, 2015.

## 2.1 PESQUISA BIBLIOGRÁFICA

Nesta fase, as bibliografias recentes da área (livros, artigos científicos e relatórios técnicos) foram buscadas *ad hoc* e estudadas. Ao final desta atividade, um relatório técnico em formato de *Survey* foi desenvolvido para ser utilizado em etapas posteriores e para possível divulgação. Na execução dessa atividade, o *Google Scholar* (2015) foi utilizado como ferramenta para buscar materiais do ano de 2010 até o presente momento. Em alguns casos, artigos mais mostraram relevância sendo citados em achados mais recentes, então, foram investigados.

## 2.2 ESTUDO COMPARATIVO

Nesta atividade, algumas soluções existentes que abordam a temática foram comparadas tendo em vista aspectos como utilidade, usabilidade e adequabilidade. Além disso, foram extraídos pontos positivos de tais abordagens para nortear o desenvolvimento da solução proposta neste projeto. Foram executadas preliminares entrevistas para avaliar algumas sugestões e alinhar a percepção sobre quais seriam as melhores propostas de estratégias e ferramentas. No apêndice A, são apresentados nos Quadro 1 e Quadro 2, material auxiliar para a realização do estudo comparativo de trabalhos acadêmicos.

## 2.3 ENTREVISTAS PRELIMINARES

Entrevistas do tipo aberta foram executadas previamente para identificar o que profissionais sentem durante o processo de desenvolvimento de *software* para integrar segurança. Além disso, as entrevistas apoiaram, na priorização das práticas identificadas na fase anterior desta pesquisa, a revisão da literatura.

### 2.3.1 Perfil profissional dos entrevistados

Foram entrevistados previamente três profissionais. O primeiro foi um especialista em construção em sistemas embarcados. Além de ser professor de segurança há dois anos, tem onze anos de experiência trabalhando com segurança em baixo nível, criptografia, infraestrutura de chaves públicas e redes sem fio.

O segundo entrevistado é um profissional multidisciplinar com mais de dez anos de experiência em tecnologia da informação. Entretanto, ele é focado em Segurança da

Informação; possui forte experiência em Testes de Penetração e Análise de Segurança. Foi analista de segurança por três anos.

O terceiro entrevistado foi mestre em *Information security and computer forensics* pela *University of East London* e teve aproximadamente 6 anos de experiência como consultor de segurança da informação. Atualmente é engenheiro de sistemas.

### 2.3.2 Dados coletados

Através das entrevistas preliminares foi detectada carência de conscientização dos envolvidos no projeto sobre a importância de segurança para a criação da cultura de preocupação e implantação de segurança. Ainda assim, é muito difícil, porque os clientes querem o sistema pronto o mais rápido possível e pagando o mínimo possível, então, atividades de segurança acabam sendo sacrificadas, até porque não são explicitadas no processo.

Outra medida interessante sugerida pelas entrevistas foi a tentativa de certificações internacionais como a *Common criteria (2015)*, para assegurar que o produto final fosse seguro. Entretanto, certificações podem ser processos pesados e caros, conflitando novamente com outros interesses.

Em adição, normalmente uma *Sprint* que tem duração média de 15 dias, e os testes são deixados para última hora, então, inserir testes de segurança robustos nesse processo é inviável, porque não há tempo hábil. Então, é necessário melhorar a gestão de tempo para que os testes de segurança ocorram da melhor forma possível.

Como mencionado, as práticas escolhidas para apresentação no guia também passaram por uma priorização desta equipe.

## 2.4 CONSTRUÇÃO DO GUIA

Nesta etapa, foi desenvolvida a solução proposta inicialmente no projeto que continha uma abordagem teórica sobre a possibilidade de haver agilidade e segurança, quais abordagens ou técnicas de segurança poderiam ser utilizadas para melhorar a qualidade do produto durante o desenvolvimento ágil de *software*. Por fim, foi proposto um guia com práticas para agregar segurança ao desenvolvimento ágil com o XP.

## 2.5 AVALIAÇÃO

Após o desenvolvimento da solução proposta, é importante que ela seja experimentada. Entrevista é uma técnica comumente utilizada em pesquisas de cunho exploratório (GIL, 2002). Dessa forma, para avaliar o guia construído na etapa anterior, foram realizadas entrevistas semi-estruturadas. Os entrevistados foram especialistas em agilidade, segurança e desenvolvimento de *software*. Para isso, uma pesquisa qualitativa foi realizada utilizando entrevista. Nesta atividade, o guia foi apresentado e avaliado por especialistas com o objetivo de estabelecer possíveis vantagens/desvantagens na sua utilização e diferenciais competitivos.

## 2.6 DOCUMENTAÇÃO

Nesta etapa, o foco foi na elaboração de documentação final relacionada ao projeto, tanto para fins de registro de informações relevantes como também para submissão do trabalho e apresentação.

# 3. REFERENCIAL TEÓRICO

Nesta secção, é apresentada o levantamento bibliográfico dos trabalhos analisados que estão relacionados ao objetivo deste trabalho. São apresentadas as abordagens e métodos que foram sugeridos nos trabalhos relacionados a este trabalho de conclusão de curso. Dessa forma, partir da revisão de trabalhos, por meio de pesquisa bibliográfica, foi possível explorar o conhecimento necessário para construção da solução proposta.

## 3.1 METODOLOGIAS ÁGEIS

As metodologias ágeis seguem um conjunto de valores e princípios que estão propostos no Manifesto Ágil (KENT et al., 2001). Existem diversos *frameworks* ágeis como o *eXtreme programming* (BECK, 2000), *Scrum* (SCHWABER, 2011), *Kanban* (ANDERSON, 2010). Dessa forma, tais princípios e valores desempenham o papel de centralizar os fundamentos ágeis para atingir melhores formas para o desenvolvimento de *software*.

Assim, os valores ágeis classificam o que é mais relevante para a perspectiva ágil. O primeiro dos quatro valores propostos pelo Manifesto Ágil postula que indivíduos e suas interações são mais importantes do que processos e ferramentas. Isto é, a comunicação de qualidade entre os *Stakeholders* e equipe é mais valioso do que processos gerais, inclusive ágeis, e também ferramentas. O segundo valor do Manifesto Ágil estabelece que *software* funcionando é mais precioso do que vasta documentação. Isso é facilmente justificável, pois documentações abrangentes são muitas vezes dispendiosas e raramente utilizadas. O terceiro valor ágil expressa a maior importância da colaboração com o cliente do que contratos rígidos, ou seja, apesar da existência de um contrato, quando necessário, é interessante negociar com cliente para melhor satisfazê-lo. O quarto valor está relacionado à necessidade de adaptação que é inerente aos negócios dinâmicos, portanto, a capacidade de resposta a mudanças é mais significativo do que seguir puramente o planejamento.

O Manifesto Ágil (KENT et al., 2001) descreve doze princípios. O primeiro princípio narra que a maior prioridade é atingir a satisfação do cliente entregando rapidamente e constantemente o *software* funcionando com os requisitos que agregam maior valor. O segundo princípio descreve que adaptação a mudanças é essencial para que o cliente tenha vantagens competitivas, e, portanto, mudanças de requisitos devem ser aceitas. O princípio seguinte trata da entrega de software funcionando de forma constante e incremental em curtos

períodos de tempo como a cada quinze dias. O quarto princípio menciona que a equipe de desenvolvimento deve trabalhar em conjunto com pessoas que estão associadas a negócios, pois isso facilita que requisitos sejam refinados, alinhados e avaliados. O quinto princípio relata a importância da motivação, indivíduos motivados executam melhor seus papéis, portanto, é proveitoso oferecer o suporte que a equipe precisa e confiar nela. O sexto princípio pronuncia que conversar pessoalmente e diretamente com as partes interessadas é a forma mais eficiente e eficaz de propagar a informação. O sétimo princípio relata que a métrica principal do progresso é o *software* funcionando. O oitavo princípio afirma que o desenvolvimento ágil é sustentável e que é possível manter um ritmo de trabalho constante para os envolvidos no projeto. O nono princípio estabelece que deve haver atenção contínua à excelência técnica e que um bom *design* melhora a agilidade. O décimo princípio aborda a necessidade de simplificar para evitar trabalhos que não precisam ser feitos para atingir o objetivo. O décimo primeiro princípio menciona que as melhores arquiteturas, *design* e requisitos são alcançados através de times auto organizáveis. O décimo segundo princípio é extremamente importante para a melhoria contínua. Ele menciona que periodicamente times ágeis devem refletir sobre como se tornar mais efetivo e se ajustar para se aperfeiçoar (KENT et al., 2001).

Como foi dito anteriormente, existem uma variedade de *frameworks* ágeis, entretanto, cada uma com suas especificidades. O conhecimento de algumas dessas particularidades irá ajudar no entendimento do guia. Uma vez que algumas recomendações fazem alusão às sugestões de outros *frameworks*: como o papel do *Scrum master* (SCHWABER, 2011) no *Scrum* ou *XP coach* (BECK, 2000) na *eXtreme Programming*, tais papéis visam manter o time focado no desenvolvimento e ajudam o time a utilizar o processo ágil em questão; O *Planning game* na metodologia XP ou *Planning poker* no *Scrum* em que o time se reúne para estimar de custo para priorização de atividades; *User Stories* (SCHWABER, 2011; BECK, 2000) que são uma forma de descrever requisitos; *Product Backlog* (SCHWABER, 2011) que são uma lista de funcionalidades priorizadas pelo cliente; *Sprint* (SCHWABER, 2011) é um intervalo curto de tempo, menor que um mês, com duração fixa, em que o time de desenvolvimento entrega um incremento do sistema; *Sprint Planning meeting* (SCHWABER, 2011) é uma reunião feita a cada *sprint* para definir quais *user stories* têm maior prioridade e o que deve ser executado na *Sprint*.

## 3.2 SEGURANÇA

O uso do computador promoveu a necessidade de proteger informações e arquivos contra acessos indevidos. Para atender essa demanda, foram desenvolvidas técnicas e ferramentas. Esse conjunto de ferramentas e técnicas projetadas com o objetivo de preservar dados é chamado genericamente de Segurança Computacional. Segurança é definida como o conjunto de medidas para proteger o sistema (IETF, 2015). Entretanto, com sistemas distribuídos, redes são utilizadas e assim, os dados deixam de estar apenas no domínio de uma máquina física, sendo necessário recorrer a segurança de rede ou segurança de internet para proteger dados (STALLINGS, 2008). O objetivo da segurança de *software*, em geral, é garantir três principais atributos: confidencialidade, integridade e disponibilidade. A confidencialidade é uma propriedade que garante a ausência da divulgação não permitida pelo proprietário da informação. A integridade é o atributo que trata da garantia de que alterações não autorizadas são inexistentes. Disponibilidade é a propriedade que está associada à prontidão do serviço ou informação fornecida pelo sistema (CORREIA; SOUSA, 2010).

A segurança é muitas vezes ignorada durante a construção de *software* por ser um requisito não funcional antagônico a outros requisitos (CORREIA; SOUSA, 2010). O mercado normalmente avalia os produtos principalmente pelas funcionalidades que o *software* oferece. *Quando* o número de funcionalidades é elevado, a complexidade do sistema também fica elevada e, portanto, a dificuldade de inserir segurança cresce, porque as práticas de segurança necessárias ficam também mais complexas, pois precisam compreender todos os casos do sistema. Outro requisito que contrasta com segurança é a usabilidade, mecanismos de autenticação, por exemplo, dificultam o acesso ao sistema. Além disso, temos questões de desempenho em que algoritmos de criptografia seriam entraves, exigindo tempo de processamento e reduzindo o desempenho do sistema. Tempo e custo são aspectos que também conflitam com segurança, visto que segurança exige tempo e custo para ser adicionada ao *software*, porém, esses recursos normalmente são limitados em projetos, e a segurança acaba sendo vista com baixa prioridade e sacrificada em detrimento de outros aspectos.

Existem alguns processos e soluções importantes para inserção de segurança: *Common criteria* (2015) é um padrão internacional para certificar segurança computacional. Uma vez tendo essa certificação o produto não precisa de avaliações adicionais. *Touchpoints* (MCGRAW, 2009), CLASP (OWASP, 2014a;) e Microsoft SDL (MICROSOFT, 2015b)

apresentam um conjunto de práticas recomendadas para o desenvolvimento de *software* seguro, entretanto, muitas das atividades sugeridas envolvem muita documentação e criam um processo burocrático e conflitante com as metodologias ágeis.

## 3.3 METODOLOGIAS ÁGEIS E SEGURANÇA

Como a indústria de *software* está utilizando as metodologias ágeis (WILLIAMS, 2010; STANDISH GROUP, 2013), é necessário prestar atenção em como requisitos de segurança estão sendo tratados, visto que há uma dificuldade inerente à execução de práticas da engenharia de segurança em projetos ágeis (BACA; CARLSSON, 2011). Portanto, algumas abordagens para mitigar esse problema são sugeridas na literatura.

Em Keramati (2008), 70 atividades de segurança foram selecionadas para o estudo, sugerindo método para adição de segurança no processo ágil de desenvolvimento com um parâmetro ajustável para o controle das características ágeis do processo. Essa análise envolve o cálculo do grau de agilidade das atividades e um parâmetro de tolerância para redução das características ágeis, ou seja, um estudo matemático sobre a redução da agilidade quando recomendações tradicionais de segurança são utilizadas. Neste trabalho, possibilita-se entender que nem todas as atividades de segurança devem ser integradas aos processos ágeis, pois isso reduziria a agilidade; não se apresenta resultados de entrevista ou avaliação.

Nesta mesma linha, no trabalho, é sugerida uma abordagem um tanto quanto semelhante à anterior (SONIA; SINGHAL; BANATI, 2014), mas com alguns diferenciais: um *framework* chamado FISA-XP e a ferramenta TISA-XP são propostos para integrar atividades de segurança com atividades com *eXtreme programming*. Para isso, atividades de segurança e atividades ágeis são comparadas, e a viabilidade da integração é verificada utilizando-se métodos matemáticos e matrizes para tal. Nesse trabalho, 30 atividades de segurança que são do CLASP (*Comprehensive lightweight application security process*) foram consideradas. Uma empresa de desenvolvimento de *software* utilizou a abordagem e a avaliou positivamente. Um guia com recomendações não foi feito.

Na proposta, o uso de práticas da Engenharia de Segurança junto aos processos ágeis é avaliado através de entrevistas com desenvolvedores (BACA; CARLSSON, 2011). Três processos de segurança foram utilizados para identificar atividades de segurança: *Microsoft SDL*, *Cigatel touchpoints* e *Common Criteria*. Entrevistados que trabalham com desenvolvimento de *software* apontaram aspectos relevantes para a integração da Engenharia

de Segurança com metodologias ágeis. As práticas de consideradas segurança não foram adaptadas ao contexto ágil.

No trabalho, é realizada uma revisão bibliográfica (GHANI; YASIN, 2013), e trabalha-se respondendo com alguns questionamentos: “Os elementos de segurança são discutidos para o XP? As pesquisas mencionam tais elementos?”, “Quantas pesquisas que mencionam elementos de segurança podem contribuir para esse estudo?”, “Existem modelos ou *frameworks* que integram XP com segurança?”, “Qual o nível de aceitação do ambiente de desenvolvimento ao XP?”

Em Pereira et al. (2011), é apresentada uma relação entre segurança com tópicos como: banco de dados relacional orientado a objetos, arquitetura multicamadas e métodos ágeis. Não foca o estudo em atividades específicas de segurança e nem em metodologia ágil. O trabalho destaca a importância para a segurança da informação a qualidade no desenvolvimento de sistemas informatizados, meta a ser atendida sem detrimento do desenvolvimento ágil.

Uma abordagem focada no Scrum é apresentada em Azham e Ghani (2011) onde princípios de segurança são tratados e uma revisão bibliográfica sobre problemas de segurança atrelados ao *Scrum* é realizada. As principais contribuições desse trabalho são: criação do *Security master* como novo papel no *Scrum* e a criação de um *Security backlog* para tratar da gestão de segurança em projetos ágeis com *Scrum*.

Um resultado de um estudo exibido em (BARTSCH, 2011), classificando os achados e aprimorando o entendimento de segurança junto às práticas ágeis, apresentando as perspectivas de profissionais de segurança no Desenvolvimento Ágil. Além disso, foi realizada entrevista com 10 praticantes de agilidade de 9 empresas. As entrevistas tratavam de pontos como o envolvimento do cliente, experiência e percepção de segurança do desenvolvedor, efeitos da agilidade na segurança, práticas de segurança e autorização.

Em Beznosov (2003), o autor propõe uma aplicação de práticas do XP para a Engenharia de Segurança e relata os benefícios. Não foca em atividades específicas de processos de segurança e não apresenta entrevista ou *feedback*.

Em proposta apresentada (WILLIAMS; MENEELY, 2010), o trabalho motiva o uso do *Protection poker*: útil para estimar riscos à segurança, adaptação da lista de requisitos à segurança, priorizar, incentivar debates com o time de desenvolvimento acarretando no compartilhamento de conhecimento sobre segurança de *software*. Nesse trabalho é apresentado um estudo de caso acompanhando um time de desenvolvimento.

A abordagem de Peeters (2008) incentiva uma extensão das *User stories* utilizadas nas metodologias ágeis para a criação de *Abuse stories*. Não foca em atividades específicas de processos de segurança tradicionais e não apresenta entrevista ou *feedback*.

Há trabalho que propõe extensão do XP e suas práticas para oferecer suporte à Engenharia de Segurança e requisitos de segurança sem perder os benefícios das metodologias ágeis (WÄYRYNEN; BODÉN; BOSTRÖM, 2005). As atividades de segurança utilizadas nesse trabalho são de padrões de segurança como o *Common criteria* e à extensão do *Planning game*. Em entrevistas, estudantes afirmaram que era mais fácil de se trabalhar com a extensão do que com o *Common criteria*.

Em GE et al. (2006), é desenvolvido um processo ágil baseado no FDD com foco na análise de risco incremental e preparo de mecanismos de segurança a cada incremento e reavaliação das ameaças.

No trabalho (BEZNOSOV; KRUCHTEN, 2004), é feita uma análise de como e quais práticas de segurança podem ser inseridas em processos ágeis.

Em outra abordagem focada no *Scrum* (ANDRADE; QUEIROZ, 2011), é desenvolvida uma extensão do *Scrum* preocupada com a segurança da informação. Abordagem semelhante é a de Azham (2011), mapeando elementos do *Backlog* focados em segurança.

Um trabalho essencial para a fundamentação deste guia foi o da Microsoft (MICROSOFT, 2015c), no qual é proposto um ciclo de desenvolvimento ágil focado em segurança. Todavia, esse trabalho é muito focado em materiais e ferramentas da Microsoft.

Em Sourcefire (2015) o *Agile security manifesto* exibe uma abordagem filosófica escrita através de 12 princípios sobre como abordar segurança em um processo adaptativo.

A partir dessa análise bibliográfica, pode-se verificar que apenas se propõe uma ferramenta (SONIA; SINGHAL; BANATI, 2014). Tal ferramenta tem o propósito de ajudar desenvolvedores a selecionar quais atividades de segurança podem ser integradas aos processos ágeis, utilizando a abordagem descrita no artigo. Então, seria interessante uma ferramenta que além de oferecer apoio a essa abordagem, suporte outras abordagens descritas nos artigos. Outro ponto, a análise do custo financeiro da inserção de requisitos de segurança em projetos ágeis não foi praticamente mencionada nos artigos encontrados nesta pesquisa. Apenas em Peeters e Dyson (2007) tal tema foi devidamente explorado. Com a pesquisa bibliográfica, foi possível encontrar fundamentos de que é possível integrar agilidade e segurança, conhecer quais abordagens e técnicas estão sendo incentivadas. Como foi visto

nesta secção, existem diversos trabalhos relacionados, dentre esses o *Microsoft SDL agile* (2015) e Bartsch (2011) são os que mais se destacam por oferecerem abordagens mais próximas a este trabalho. É importante destacar que este trabalho se diferencia dos outros por apresentar, de forma amigável, estratégias simples para integração entre agilidade e segurança. Além de apresentar uma discussão profunda sobre as vantagens e desvantagens e avaliação das recomendações.

# 4. DESENVOLVIMENTO DO GUIA

## 4.1 ESTRUTURAÇÃO DO GUIA

Uma vez que a segurança é comumente negligenciada durante o desenvolvimento ágil de software (KERAMATI, 2008), foi feita uma revisão exploratória da literatura e entrevistas com especialistas com o objetivo de identificar um conjunto de práticas primárias para serem incorporadas ao processo de desenvolvimento de *software* ágil a fim de aprimorar o nível de segurança. Nesse sentido, este trabalho apresenta um guia para suporte à inserção de requisitos e práticas de Engenharia de Segurança em projetos ágeis.

As atividades de segurança são propostas através do formato de *Guidelines* (SOARES, 2009) cujo formato segue o padrão detalhado abaixo:

- **Título:** Nomenclatura utilizada para identificar a prática sugerida no *Guideline;*
- **Descrição:** Breve introdução e explicação conceitual sobre a prática;
- **Benefícios:** A melhoria causada através da adoção da prática;
- **Trabalhos relacionados:** Quem recomenda o uso da prática;
- **Passos:** Quando necessário, orientações mais detalhadas para executar a prática.

Assim como as metodologias ágeis que apesar de serem fundamentadas em princípios e valores semelhantes, elas têm variações nas abordagens (exemplo: *Scrum*, *XP* e *Kanban*, entre outras). As sugestões do guia são baseadas na revisão da literatura, entretanto, algumas práticas são similares e recomendadas por mais de um trabalho relacionado com pequenas variações. Nesse contexto, este guia propõe atividades de forma particular embasado na literatura e em conversas com profissionais. A adoção das sugestões do guia pode ser total ou parcial, dependendo das necessidades e contexto do projeto.

## 4.2 GUIA PARA SUPORTE A INSERÇÃO DE REQUISITOS DE SEGURANÇA EM PROJETOS ÁGEIS

### *4.2.1 Security awareness training*

#### 4.2.1.1 Descrição

Essa prática tem a intenção de preparar, conscientizar e responsabilizar os envolvidos no projeto sobre a importância da segurança. Durante o treinamento, é essencial abordar

tópicos como a política de privacidade; conceitos básicos, principais ataques, estratégias de defesa e modelagem de ameaças. O melhor momento para introduzir o treinamento é no início do projeto, podendo ser reforçado periodicamente para contribuir para a excelência técnica e conscientização.

4.2.1.2 Benefícios

A fase de treinamento prepara e reforça os membros do projeto para lidar efetivamente com os requisitos de segurança.

4.2.1.3 Trabalhos relacionados

Os da Microsoft (2015) e Owasp (2014).

### *4.2.2 Security master*

4.2.2.1 Descrição

Determinar uma comissão ou alguém com *expertise* em segurança para direcionar e motivar segurança no ciclo de desenvolvimento. Assim como o *Scrum master* (SCHWABER, 2011) é uma espécie de guardião do *Framework* ágil, o *Security master* é um papel que deve gerenciar e assegurar que as atividades de segurança serão implementadas. A entidade do *Security master* deve liderar treinamentos de segurança, conscientizar o time a respeito dos requisitos de segurança; liderar a modelagem de ameaças, planejamento e objetivos de segurança a serem seguidos.

Os membros do time são responsáveis por executarem as atividades de segurança satisfatoriamente. O *Security master* deve funcionar como um repositório de conhecimento de segurança e oferecer consultoria. Quando necessário, ele também deve revisar e avaliar o trabalho dos membros do time, realizar testes de segurança, identificar riscos de segurança e auxiliar na construção de estratégia. Se o time participou do *Security awareness training*, a necessidade de um *Scrum master* alocado exclusivamente ao projeto pode diminuir, pois o time já está apto a lidar com aspectos de segurança. Assim, uma empresa com muitos projetos pode ter uma comissão de *Scrum masters*. Assim, eles oferecem consultoria a diversos projetos, avaliando e executando tarefas mais específicas. O uso de uma comissão é interessante, pois o profissional será aproveitado em diversos projetos dentro de uma empresa, fazendo o custo de um *Security master* cair.

4.2.2.2 Benefícios

Liderança técnica para atividades de segurança, repositório de conhecimento sobre segurança e consultoria de segurança.

4.2.2.3 Trabalhos relacionados

Os de Azham e Ghani (2011); Owasp (2014); Microsoft (2015); Andrade (2011); Wäyrynen (2004).

### *4.2.3 Security backlog*

4.2.3.1 Descrição

O *Product backlog* (SCHWABER, 2011) é uma lista de atividades que geralmente é aprovada pelo cliente. Entretanto, usuários e clientes geralmente não possuem conhecimento profundo sobre riscos de segurança. Dessa forma, o *Security master* pode gerenciar a lista de atividades de segurança que tem o nome de *Security backlog*. A partir do *Product backlog*, o *Security master* deve identificar quais vulnerabilidades podem estar associadas às atividades e documentar quais funcionalidades precisam de atividades de segurança. Dessa forma, *Security backlog* é uma lista de atividades, assim como o *Product backlog*. Assim, funcionando como um planejamento das atividades de segurança incluindo revisão de código e testes de aceitação.

Dessa forma, de acordo com as prioridades, são montadas as *Sprint backlogs*, levando em consideração o *Security backlog*. Caso seja mais confortável para o time, elementos de segurança podem ser pontuados diretamente no *Product Backlog*, o importante é antecipar a preocupação com segurança desde o planejamento.

4.2.3.2 Benefícios

A preocupação com segurança é antecipada e evidenciada. Além disso, a visibilidade das atividades facilita a rastreabilidade dos requisitos de segurança.

4.2.3.3 Trabalhos relacionados

Os de Azham e Ghani (2011) e Andrade (2011).

4.2.3.4 Passos

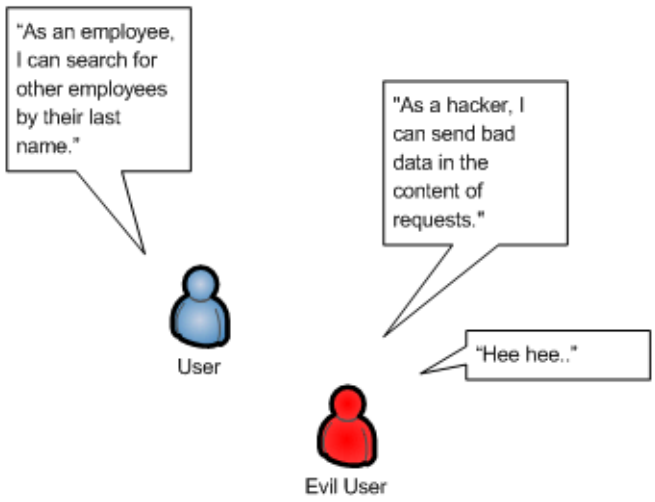
Analisar atividades do *Product backlog* para identificar possíveis vulnerabilidades, documentar as possíveis vulnerabilidades associadas, criando o *Security backlog* para posteriormente o time tratar a implementação e testes de segurança.

### *4.2.4 Evil user story*

4.2.4.1 Descrição

Nas metodologias ágeis é comum o uso de *User stories* (SCHWABER, 2011) que são descrições informais dos requisitos escritos pelo *Product owner*, as *User stories* relatam o objetivo para o requisito em questão, quem precisa e qual o benefício. As *Evil user stories* são baseadas nas *User stories*, entretanto, o *Security master* ou alguém com conhecimento técnico deve escrevê-las, pois eles estão aptos a perceberem com mais facilidade, vulnerabilidades nos cenários a serem analisados do que pessoas não técnicas. Mas ainda assim, é importante que o *Product owner* participe, pois ele conhece das regras de negócio, podendo indicar, por exemplo, quais dados são mais sigilosos e quais precisam de maior proteção. Ou seja, diferentes perspectivas são importantes para escrever. O objetivo é hackear o *Product backlog* criando *User stories* como se um *hacker* ou usuário mal intencionado estivesse planejando uso inadequado do sistema. Não é necessário criar *Evil user stories* para todas as histórias, apenas para aquelas que podem apresentar algum tipo de vulnerabilidade. A Figura 3 representa um exemplo de Evil user story.

Figura 3 – *Example Evil user story*




Fonte: Owasp (2011).

4.2.4.2 Benefícios

Detectar possíveis falhas de segurança e destacar a necessidade de contramedidas para histórias mal intencionadas. É importante salientar que a modelagem dessas possíveis ameaças é feita baseada na arquitetura *baseline* do sistema e pode ser modificada com as alterações do sistema.

4.2.4.3 Trabalhos relacionados

Os de Owasp (2011); Microsoft (2015); Peeters (2005); Mcgraw (2009).

4.2.4.4 Passos

Identificar os *Stakeholders* necessários para identificar como podem ameaçar a segurança do sistema. A equipe de desenvolvimento contribuindo com o aspecto técnico e os stakeholders com a compreensão do negócio podem detectar e documentar através das *Evil user stories* as vulnerabilidades presentes nos cenários e por fim tratá-las nas *Sprints*.

### *4.2.5 Protection poker*

4.2.5.1 Descrição

É uma técnica em que os participantes interagem e se comunicam para chegar ao melhor resultado possível em consenso com as atividades de segurança e estimam o risco associadas às atividades a serem executadas na *Sprint*.

4.2.5.2 Benefícios

Envolve a visão de diversos membros do time sobre segurança. Os envolvidos podem apresentar diferentes pontos de vista e compartilhar experiências. Essa discussão também aumenta o conhecimento sobre segurança dos envolvidos, ajudando na criação de estratégias além de estimar os riscos de segurança.

4.2.5.3 Trabalhos relacionados

Os de Willians e Meneeky (2010); Boström (2006).

4.2.5.4 Passos

Durante a sessão do *Planning meeting* que acontece a cada iteração, o *Product owner* (SCHWABER, 2011) explica os requisitos para o time com o objetivo de esclarecer as dúvidas a respeito do dos itens da *Sprint backlog* em questão. Com os requisitos refinados, o time discute sobre as implicações de segurança atrelados aos requisitos. A discussão pode levantar questões como: “Quem desejaria atacar o sistema?”, “Quais as vulnerabilidades?”, “O que o atacante poderia fazer?”, “Quais os danos acarretados pelo acesso indevido?”, “Qual estratégia para proteger?”. Como já estamos propondo o *Security backlog*, *Evil user stories* & *Abuse stories* alguns desses questionamentos já estarão bem direcionados e serão revisados e discutidos. Quando a discussão cessar, o time vota para expressar a estimativa assim como no *Planning poker*. Entretanto, quanto mais alto o valor da estimativa, maior o risco de segurança

associado ao requisito. A estimativa é expressa de acordo com a sequência de Fibonacci, assim como no *Planning poker* (SCHWABER, 2011).

### *4.2.6 Quality gate*

4.2.6.1 Descrição

São importantes para garantir a qualidade do produto. *Quality gates* definem o grau mínimo de privacidade e segurança aceitáveis em *releases.* Um *Quality gate* é um acordo estabelecido pelo time para que o produto final tenha qualidade. São exemplos, nenhuma falha de segurança crítica deverá estar presente na *release* ou a implementação deve obedecer às políticas de segurança X, Y e Z.

4.2.6.2 Benefício

Reforça a atenção voltada para falhas de segurança do sistema; reduz a possibilidade de que a *release* contenha tais problemas e estabeleça padrões de qualidade.

4.2.6.3 Trabalhos relacionados

Os da Microsoft (2015); Owasp (2014).

### *4.2.7 Incident plan*

4.2.7.1 Descrição

Planos de incidência são utilizados para identificar e planejar como responder a possíveis ameaças que possam acontecer nas *releases*. Esse planejamento envolve definição de contatos emergenciais para resoluções de segurança; definir time apto a executar do plano de incidência, nesse time deve haver alguém para tomada de decisões que entenda o impacto delas para o sistema; além disso, é importante destacar que o time deve conter pessoas capacitadas tecnicamente e preparadas para agir de acordo com o plano. Também faz parte do plano de incidência prover planejamento de serviços de segurança para código de terceiros ou já existentes no processo ou organização. Em adição, o *Incident plan* deve especificar como proceder diante de atividades suspeitas, a criação desse plano deverá ser feita antes das *releases*. Para isso, é preciso ter total conhecimento do negócio e do impacto das decisões tomadas. Durante o procedimento, é necessário ter cuidado para não adulterar evidências. Faz parte da manutenção do sistema atualizar esse planejamento.

Como exemplo da execução de um *Incident plan*: Um sistema bancário detectou uma atividade suspeita, pois um usuário utilizou o cartão de crédito em uma cidade X e, em menos

de uma hora depois, a numeração do cartão foi utilizada para fazer uma compra *online* em uma cidade Y que fica a 3500 km da cidade X. Ou seja, é estranho que a compra tenha sido efetuada pela mesma pessoa. Nesse caso, o sistema detectou uma atividade suspeita e o time responsável por executar o *Incident plan* deve atuar, entrar em contato com o proprietário do cartão de crédito para esclarecer a situação, saber se a compra foi de fato realizada pelo cliente e, se necessário, iniciar o procedimento para solucionar o ocorrido. Para resolver ou mitigar a situação, talvez seja necessário estornar a compra, cancelar temporariamente o cartão, enviar outro cartão para o cliente com diferente numeração. Todo o processo de recuperação é planejado no *Incident plan*.

#### 4.2.7.2 Benefícios

O planejamento torna a entidade mais bem preparada para aplicar contramedidas, fazendo com que a recuperação ocorra mais rapidamente e de forma eficiente com impacto negativo mitigado.

#### 4.2.7.3 Trabalhos relacionados

Os da Microsoft (2015); Berkeley (2015).

#### 4.2.7.4 Passos

Depois de definir o planejamento, como foi descrito anteriormente, para executar um *Incident plan*, geralmente as seguintes fases ocorrem: detecção, análise, recuperação e pós-incidente. A detecção é a fase em que ocorre a percepção da ocorrência e classificação da prioridade. A segunda fase é a de análise. Nela, as atividades adicionais de resposta são definidas, avaliadas em termos de impacto e priorizadas. Na recuperação, são executadas atividades de contenção e erradicação dos efeitos negativos para recuperação do sistema. Algumas vezes é imprescindível executar as fases de detecção e análise para certificar que o sistema de fato está recuperado ou que não há outra ameaça/falha ocorrendo em outra instância. Depois que o incidente foi contornado, a fase pós-incidente tratou de reportar a causa e o efeito do incidente e o que deveria ser feito para evitar outras ocorrências futuras.

### *4.2.8 Research vulnerabilities*

4.2.8.1 Definição

Muitas vulnerabilidades são descobertas ao longo do tempo, então, para se proteger é necessário pesquisar possíveis vulnerabilidades associadas ao projeto para aplicar contramedidas. Alguns sites como Mitre Corporation (2015), Offensive Security (2015) têm um acervo atualizado de vulnerabilidades descobertas. A execução dessa pesquisa chama atenção do time para possíveis vulnerabilidades. A pesquisa por vulnerabilidades pode envolver a atualização das ferramentas, pois em alguns *updates*, correções de segurança são feitas. Por exemplo, a Microsoft Security Response Center fornece um boletim de segurança para documentar mensalmente as atualizações de segurança disponibilizadas através de *updates* (MICROSOFT, 2015). Por isso, muitas vezes é importante atualizar as ferramentas para prover maior segurança. Vale salientar que se as atualizações das ferramentas forem feitas com certa frequência e de forma incremental, a possibilidade de grandes transtornos para migração de versões tende a ser menor e as mudanças menos agressivas.

4.2.8.2 Benefícios

Estar atualizado com as vulnerabilidades que podem estar associadas ao projeto para poder aplicar contramedidas.

4.2.8.3 Trabalhos relacionados

Microsoft (2015); Owasp (2015).

4.2.8.4 Passos

Caso, através da pesquisa, vulnerabilidades tenham sido encontradas, as contramedidas devem ser aplicadas. Essa pesquisa deve ser feita periodicamente para evitar que o produto permaneça vulnerável por um período longo de tempo.

### *4.2.9 Security verifications*

4.2.9.1 Definição

Três verificações de segurança são fortemente recomendadas:

- **Static analysis:** Acontece durante a fase de desenvolvimento. Antes de o código ser compilado, é feita a análise do código-fonte para detectar possíveis vulnerabilidades automaticamente através de ferramentas.

- **Dynamic analysis:** É executado na fase de testes. Desempenha verificações em tempo de execução, usando ferramentas que monitoram aplicações. Existem duas categorias principais para a análise dinâmica: *Vulnerability scanning*, *Penetration testing*. A varredura de vulnerabilidades emprega *software* que procura falhas de segurança com base em um banco de dados de falhas conhecidas, sistemas de teste para a ocorrência destas falhas, gerando um relatório das conclusões que um indivíduo ou uma empresa pode usar para aumentar a segurança da rede. O teste de penetração é um método que avalia a segurança de um sistema de computador ou de uma rede, simulando um ataque de uma fonte maliciosa. O processo envolve uma análise nas atividades do sistema que envolve a busca de alguma vulnerabilidade em potencial.

- **Fuzzy test:** Testar o programa, introduzindo automaticamente dados semi-aleatórios (números, *strings*, comandos SQL) para detectar falhas de segurança através da análise do impacto causado. É um teste caixa preta, ou seja, o sistema é avaliado pelos resultados obtidos através das entradas.

4.2.9.2 Benefícios

Através das verificações de segurança, é possível identificar vulnerabilidades e falhas a fim de aplicar as contramedidas necessárias.

4.2.9.3 Trabalhos relacionados

Os de Owasp (2015); Microsoft (2015); Mcgraw (2009); Arkin (2005); Chess (2004).

# 5. AVALIAÇÃO

A avaliação deste trabalho ocorreu através de entrevistas para fins de pesquisa qualitativa. Seis entrevistados com diferentes qualificações profissionais foram questionados a respeito do guia proposto neste trabalho e suas práticas. Cada entrevista é apresentada com a seguinte estrutura: a primeira sessão identifica o perfil do profissional entrevistado; na segunda sessão são apresentados os comentários relevantes feitos durante a entrevista. Além do *feedback* sobre as recomendações do guia, os entrevistados foram pedidos para classificarem numa escala de 0 a 10 a utilidade do guia, 0 inútil e 10 muito útil; Os entrevistados também foram solicitados a definir numa escala de 0 a 10, a facilidade de uso do guia, 0 sendo difícil e 10 muito fácil. É possível observar o resultado desses dados na Tabela 1. Utilizando essa mesma escala, os entrevistados também foram convocados a avaliar cada prática do guia isoladamente a respeito da utilidade. Os resultados podem ser observados na Tabela 2. Semelhantemente, os resultados sobre a facilidade de uso de cada recomendação podem ser observados na Tabela 3.

## 5.1 ENTREVISTA 1

### 5.1.1 Perfil profissional do entrevistado

O entrevistado exerce a atividade de analista e desenvolvedor de sistema há três anos, atualmente é líder técnico de uma *start up* que conta com uma equipe de desenvolvimento de *software* com sete pessoas, distribuídas entre *mobile*, *front-end*, *back-end* e *designer*.

### 5.1.2 Particularidades da entrevista

A técnica *Security awareness training* foi considerada de extrema importância, pois o entrevistado afirma que grande parte das negligências de segurança cometidas por desenvolvedores são ocasionadas por falta de conhecimento sobre um tipo de ataque ou sobre vulnerabilidades.

O *Security master* para o entrevistado talvez seja a melhor sugestão do guia, visto que a presença de uma pessoa experiente e dedicada a garantir a segurança do sistema vai implicar melhorias para segurança e difusão do seu conhecimento entre os membros da equipe.

O entrevistado acha válido inserir comentários e observações sobre segurança no *Product backlog*. Assim toda a equipe toma conhecimento dos possíveis problemas de

segurança que podem ocorrer em cada atividade, porém, teme que o processo fique burocrático.

Foi considerado importante definir através dos *Quality gates* quais são os níveis de qualidade mínimos e suas prioridades. Contudo, é possível que isso ocasione certa negligência sobre alguns aspectos de baixa prioridade. Por exemplo, se é estabelecido que nenhuma falha crítica de segurança deve estar presente numa *release*, então, as que não estão definidas no acordo podem passar? Pelo bom senso, a resposta é não.

## 5.2 ENTREVISTA 2

### 5.2.1 Perfil profissional do entrevistado

O entrevistado é especialista em agilidade e tem dezessete anos de experiência trabalhando com tecnologia da informação. Atualmente é gerente de projetos numa empresa com aproximadamente quinhentos funcionários.

O entrevistado tem os seguintes certificados: Project Management Professional (PMP)®, Certified Scrum Master - CSM®, Certified Scrum Product Owner - CSPO®, Professional Scrum Master - PSM I®, Implementador P2 MPS.BR - Melhoria de Processo de Software Brasileiro, ITIL Foundation, PMI-ACP® (Agile Certified Practitioneer), CSP® - Certified Scrum Professional.

### 5.2.2 Particularidades da entrevista

O entrevistado inicia a entrevista afirmando que se segurança é uma prioridade para a aplicação, ela será priorizada. Destaca que os métodos ágeis não excluem segurança do processo. Apenas é necessário conscientizar o time ágil a priorizar atividades de segurança.

Sobre o *Training security awareness* o entrevistado afirma que é extremamente útil capacitar os profissionais para tornar aplicações seguras e também que é essencial criar reforços dessa conscientização frequentemente.

De acordo com o entrevistado, a quantidade de profissionais com conhecimento em segurança para executar o papel do *Security master* é escassa. Então o *Security master* é útil, entretanto pode custar muito caro ficando alocado em um único projeto. Então, compartilhá-lo entre projetos é uma boa opção, porém, gerenciar as atividades dele quando alocado em vários projetos é complexo. Para driblar isso, uma sugestão foi ter alguém (dependendo da necessidade mais de uma pessoa) na organização como *Security master* para consultorias de

segurança. No cenário em que o *Scrum master* do projeto teve o *Security awareness training*, ele poderia alertar o time de que o projeto precisaria de procedimentos de segurança, verificando se os mesmos estão sendo feitos. Para *releases*, um especialista poderia fazer auditorias periódicas para ajustes mais específicos de segurança, fornecendo uma espécie de suporte.

Para a criação do *Security backlog* o entrevistado afirmou que as atividades de segurança irão ser criadas naturalmente à medida que o entendimento do produto vai sendo refinado. O entrevistado propõe que revisões de segurança ocorram durante o processo de desenvolvimento e que as questões de segurança sejam tratadas quando os problemas forem identificados e não previamente. Do ponto de vista de segurança, o entrevistado considera estranho, porém, partindo do princípio de que houve um treinamento de segurança antes e que as pessoas estão aptas a criar um produto seguro.

Sobre as *Evil user stories*, o entrevistado afirmou gostar da prática, porém, sugeriu unificar com a *Protection poker* já que as histórias podem surgir naturalmente dentro das discussões da reunião. A *Protection poker* é uma prática mais completa.

A respeito de criar *Quality gates*, o entrevistado afirma que a prática pode ser feita para qualquer outro *Bug* ou padrão de qualidade e não especificamente para questões relacionadas à segurança. A utilidade da prática seria explicitar e forçar a criação de um padrão de qualidade para segurança. Entretanto, ela foi considerada extremamente fácil de ser integrada ao processo, visto que já é algo comum.

Quanto ao *Incidente plan*, afirmou que é difícil de implementar. Porém. o entrevistado mostrou-se totalmente de acordo com a prática.

A prática *Research vulnerabilities* foi considerada relevante e útil, entretanto, há uma dificuldade inerente ao tratamento das vulnerabilidades encontradas. A facilidade de executar a prática pode variar dependendo da contramedida necessária.

Sobre as *Security verifications*, as mesmas são bem aceitas, afinal, servem como direcionamento para aplicação das contramedidas.

No final da entrevista, o entrevistado afirmou que a apresentação do guia fez com que ele considerasse a adoção de algumas práticas nos projetos que ele gerencia.

## 5.3 ENTREVISTA 3

### 5.3.1 Perfil profissional do entrevistado

O profissional trabalha como engenheiro de sistemas há 13 anos e atualmente é líder técnico em uma empresa com aproximadamente quinhentos funcionários.

### 5.3.2 Particularidades da entrevista

O entrevistado foi receptivo ao *Security awereness training* e afirmou que é fundamental instruir, porém, implantar a consciência sobre segurança é difícil por falta de incentivo.

Sobre o *Security master*, afirma que seria interessante a nível de consultoria, por justificativas semelhantes ao do segundo entrevistado.

O *Security backlog* foi reprovado pelo entrevistado. Ele não acha ágil. Tão somente acredita que vai criar mais processo para algo que é feito para ser simples.

Sobre as *Evil user stories*, a recomendação é que os engenheiros de testes criem tais *Stories* no formato de casos de testes. O *Protection poker* foi avaliado como útil, porém, burocrático.

## 5.4 ENTREVISTA 4

### 5.4.1 Perfil profissional do entrevistado

Profissional multidisciplinar. Trabalha com Engenharia de Sistemas há vinte e três anos. Oferece consultoria entre os projetos de uma empresa com aproximadamente quinhentos funcionários.

### 5.4.2 Particularidades da entrevista

A respeito do *Security awareness training*, o entrevistado afirmou que é importante e que há maneiras de lidar com isso. A primeira é oferecendo um treinamento completo presencial. Entretanto, talvez essa não seja a maneira ideal. O entrevistado cita uma experiência que teve sobre treinamentos. Quando ele aprendeu sobre os padrões de projetos, acreditou que a qualidade do código dele melhorou bastante e sentiu a necessidade de ensinar. Então resolveu fazer grandes treinamentos na organização. Todavia, após os treinamentos, observou que a postura dos desenvolvedores continuava praticamente igual. Entretanto, um

projeto na empresa estava demandando daquele treinamento, porque houve uma série de problemas relacionados ao *design*. Dessa forma, o treinamento foi necessário naquele projeto, e os resultados obtidos pelo treinamento foram positivos. As pessoas utilizaram os conhecimentos obtidos através do curso no projeto. Baseado nessa experiência, o entrevistado propõe que uma boa estratégia é gravar o treinamento de segurança em vídeos e deixá-los disponíveis, a fim de que eles sejam utilizados pela demanda. Um bom curso depende do momento em que ele está sendo aplicado.

Quanto ao papel do *Security master*, o entrevistado apontou que essa deve ser uma medida transitória e corretiva, porque o ideal é que naturalmente os engenheiros de testes façam as verificações de segurança e que os desenvolvedores implementem as medidas de segurança. Contudo, geralmente segurança é negligenciada. Então por isso o *Security master* é necessário até que a cultura de cuidados com segurança passe a ser normal. O entrevistado afirmou também que profissionais com as qualificações necessárias para ser um *Security master* são difíceis de serem encontrados.

O *Security Backlog* foi avaliado com baixa utilidade. Primeiramente porque existem outros requisitos que são negligenciados. Então, criar um *backlog* para segurança poderia justificar a criação de vários *backlogs*. A sugestão foi que testes de segurança sejam reforçados para que atividades de segurança sejam criadas.

Sobre as *Evil user stories*, a recomendação do entrevistado é que devam ser colocadas no formato de testes de aceitação. Apesar disso, geralmente não são tratadas da forma sugerida; tais testes são negligenciados. Então, como medida corretiva, e para chamar a atenção da necessidade, as *Evil user stories* são uma alternativa, lembrando que o *framework* já suporta isso. O *Protection poker*, para o entrevistado, não é válido, porque o *framework* ágil já oferece. Por exemplo, no *Planning meeting 2* do *Scrum*, o fórum é onde as discussões sobre quaisquer estratégias devem ocorrer e seguir o processo normalmente. Nesse sentido, os *Quality gates* devem ser incorporados ao processo através de critérios de aceitação. Novamente, o *framework* ágil deve apoiar a prática, porém, não estão sendo utilizados adequadamente para segurança. Nesses casos, o entrevistado recomenda educar e não engessar o processo. Talvez essas práticas sejam até educativas, mas a longo prazo, a intenção é que a cultura mude e que as práticas não sejam necessárias.

Acerca de *Security verifications*, o entrevistado pontuou que gostaria de ser capaz de criar de forma descomplicada suas próprias regras de verificação para testes de segurança estáticos.

## 5.5 ENTREVISTA 5

### 5.5.1 Perfil profissional do entrevistado

O entrevistado é um profissional multidisciplinar com mais de dez anos de experiência em tecnologia da informação. Entretanto, focado em segurança da informação. Forte experiência em Testes de Penetração e Análise de Segurança. Foi analista de segurança por três anos. Atualmente, é engenheiro de sistemas numa empresa com aproximadamente quinhentos funcionários.

### 5.5.2 Particularidades da entrevista

Acerca de *Security awareness training*, o entrevistado afirma que um treinamento que agrega valor para segurança é custoso e leva tempo. Então, a sugestão é fazer o treinamento de acordo com a demanda do projeto.

Um ponto relevante levantado nessa entrevista foi sobre a sobrecarga do *Security master* caso ele esteja alocado em mais de um projeto. Outro ponto levantado durante esta entrevista foi que os profissionais de teste normalmente não têm as habilidades necessárias para executar os testes de segurança efetivamente. Os demais aspectos de certa forma foram apresentados nas outras entrevistas.

## 5.6 ENTREVISTA 6

### 5.6.1 Perfil profissional do entrevistado

Mestre em *Information Security and Computer Forensics* pela *University of East London*. Teve aproximadamente seis anos de experiência como consultor de segurança da informação. Atualmente é engenheiro de sistemas numa empresa com aproximadamente quinhentos funcionários.

### 5.6.2 Particularidades da entrevista

O entrevistado se mostrou bastante interessado na maioria das práticas, avaliando a maioria como construtivas para o desenvolvimento do *software* seguro. Durante a entrevista, o entrevistado relatou que tentou convencer clientes e gerentes sobre a importância de construir *software* de forma mais segura, porém, eles estão preocupados com o produto funcionando sem empregar muito tempo e mais custo.

Nessa entrevista, basicamente os aspectos sobre as dificuldades de executar as práticas propostas foram técnicos e de conflitos de interesses. A respeito do *Protection poker*, o entrevistado acredita que a discussão deva ocorrer entre especialistas em segurança ou na ausência do *Security master* para evitar que todos os resultados sejam influenciados principalmente por ele. Sobre *Security verifications*, o entrevistado questionou a confiabilidade das ferramentas de análise estática do código.

## 5.7 RESUMO DOS RESULTADOS DA AVALIAÇÃO

A Tabela 1 apresenta, em resumo, a nota que foi dada por cada entrevistado ao guia, apresentado sobre a utilidade e facilidade do guia. Já a Tabela 2 apresenta para cada prática sugerida pelo guia o conceito dado pelo entrevistado para a utilidade. O conceito atribuído seguiu uma escala, sendo “0” (zero), ruim ou pouco relevante, a “10” (dez), excelente ou muito relevante. Na tabela 3, são exibidos os conceitos dados pelos entrevistados acerca da facilidade de aplicação para cada recomendação sugerida pelo guia. O conceito atribuído seguiu uma escala, sendo “0” (zero), difícil ou complicado, a “10” (dez), simples ou descomplicado.

Para as entrevistas foram escolhidos profissionais especialistas em agilidade, segurança e que têm conhecimentos práticos sobre a execução de projetos. A escolha de profissionais com perfis diferentes foi feita para garantir que o resultado não fosse tendencioso. Durante a fase de entrevistas, foi possível perceber que especialistas em segurança eram muito mais receptivos as recomendações do guia do que outros profissionais. Com base nos dados apresentados na Tabela 1, a utilidade das sugestões apresentadas no guia teve média 8. A média não foi considerada maior porque os entrevistados afirmavam que Softwares mais simples não necessitam de elevada preocupação com segurança, mas ainda assim eram sugestões extremamente úteis para contextos onde segurança é relevante.

Ainda analisando os dados apresentados na Tabela 1, a questão facilidade teve uma média baixa. A maioria dos entrevistados falavam que a dificuldade estava no contexto de criação da maioria dos softwares: existe uma pressão para construir software rapidamente e com baixo custo. Assim, as práticas para inserção de segurança sejam sacrificadas, pois exigem tempo e com isso também há um custo associado.

Na entrevista 2, o entrevistado afirma que se segurança é uma prioridade para a aplicação, ela será priorizada no contexto ágil. Destaca que os métodos ágeis não excluem

segurança do processo. Apenas é necessário conscientizar o time ágil a priorizar atividades de segurança. Dessa forma, o entrevistado concorda que ainda assim as atividades de segurança são negligenciadas. Então as recomendações do guia ajudam a solucionar este problema.

Tabela 1 – Utilidade e facilidade do guia

| ENTREVISTA | UTILIDADE | FACILIDADE |
|---|---|---|
| Entrevista 1 | 9 | 8 |
| Entrevista 2 | 7 | 4 |
| Entrevista 3 | 7 | 4 |
| Entrevista 4 | 7 | 6 |
| Entrevista 5 | 8 | 6 |
| Entrevista 6 | 10 | 6 |

Fonte: Arquivo da autora, 2015.

Tabela 2 – Utilidade de cada recomendação

(continua)

| ENTREVISTA | TRAINING SECURITY AWARENESS | SECURITY MASTER | SECURITY BACKLOG | EVIL USER STORIES | PROTECTION POKER |
|---|---|---|---|---|---|
| Entrevista 1 | 10 | 8 | 10 | 8 | 7 |
| Entrevista 2 | 10 | 7 | 7 | 8 | 8 |
| Entrevista 3 | 8 | 9 | 7 | 7 | 6 |
| Entrevista 4 | 9 | 8 | 2 | 9 | 0 |
| Entrevista 5 | 8 | 10 | 7 | 8 | 9 |
| Entrevista 6 | 10 | 10 | 10 | 10 | 4 |

(conclusão)

| ENTREVISTA | QUALITY GATE | INCIDENT PLAN | RESEARCH VULNERABILITIES | SECURITY VERIFICATIONS |
|---|---|---|---|---|
| Entrevista 1 | 6 | 10 | 8 | 8 |
| Entrevista 2 | 0 | 10 | 10 | 9 |
| Entrevista 3 | 8 | 8 | 6 | 8 |
| Entrevista 4 | 9 | 9 | 10 | 8 |
| Entrevista 5 | 10 | 2 | 10 | 8 |
| Entrevista 6 | 10 | 10 | 10 | 10 |

Fonte: Arquivo da autora, 2015.

Tabela 3 – Facilidade de uso de cada recomendação

(continua)

| ENTREVISTA | TRAINING SECURITY AWARENESS | SECURITY MASTER | SECURITY BACKLOG | EVIL USER STORIES | PROTECTION POKER |
|---|---|---|---|---|---|
| Entrevista 1 | 7 | 6 | 9 | 8 | 9 |
| Entrevista 2 | 10 | 3 | 3 | 8 | 8 |
| Entrevista 3 | 8 | 9 | 7 | 7 | 6 |
| Entrevista 4 | 5 | 9 | 8 | 8 | 0 |
| Entrevista 5 | 2 | 5 | 8 | 7 | 7 |
| Entrevista 6 | 10 | 4 | 5 | 10 | 10 |

(conclusão)

| ENTREVISTA | QUALITY GATE | INCIDENT PLAN | RESEARCH VULNERABILITIES | SECURITY VERIFICATIONS |
|---|---|---|---|---|
| Entrevista 1 | 9 | 6 | 9 | 7 |
| Entrevista 2 | 10 | 4 | 4 | 4 |
| Entrevista 3 | 8 | 2 | 6 | 5 |
| Entrevista 4 | 6 | 3 | 5 | 8 |
| Entrevista 5 | 10 | 2 | 8 | 6 |
| Entrevista 6 | 3 | 10 | 10 | 7 |

Fonte: Arquivo da autora, 2015.

# 6. CONSIDERAÇÕES FINAIS

Projetos desenvolvidos sob as metodologias ágeis estão sendo amplamente adotados pelas organizações. Projetos ágeis são beneficiados com fatores que estão associados ao sucesso de projetos. Entretanto, para serem mais adaptativos, os projetos ágeis geralmente têm, no início, o entendimento incompleto do que deverá ser feito ao longo do projeto. Isso dificulta que as práticas de Engenharia de Segurança sejam executadas, porque elas foram preparadas para metodologias tradicionais. Portanto, um guia com um conjunto de recomendações simples para a integração das metodologias ágeis e requisitos de segurança foi construído neste trabalho. O guia foi apresentado a especialistas e avaliado como útil, mas não fácil devido ao contexto dos projetos.

Nesse sentido, várias sugestões fundamentadas na análise bibliográfica foram feitas através do guia. Uma das recomendações mais bem sucedidas deste guia é a *Security awareness training*. Essa prática foi uma das mais bem avaliadas como útil durante as entrevistas de avaliação, identificando a carência de capacitação e conscientização a respeito de segurança. Especialistas viram como uma boa ideia ter um *Security master* recomendado pelo guia, porém, relatam escassez de profissionais especialistas em segurança para ocupar o papel de *Security master*.

Em alguns casos, o *framework* ágil já suporta algumas práticas para melhoria dos níveis de segurança na construção do *software*, como o exemplo das *Evil user stories* que podem ser colocadas no formato dos já existentes casos de teste. Nesse contexto, além dos requisitos conflitantes e prioridades, foram identificados outros dois motivos para os requisitos de segurança estivessem sendo negligenciados. A primeira razão foi a falta de conscientização e motivação sobre a necessidade de segurança. Então, o uso de práticas para evidenciar os requisitos de segurança é importante até para fins educativos. O segundo motivo foi que os profissionais não estão qualificados o suficiente para lidar efetivamente com requisitos de segurança. Então, a prática do *Security awareness training* mais uma vez se justifica.

Para trabalhos futuros, as entrevistas utilizadas para avaliação do guia podem servir como auxilio para melhorias das recomendações do guia, e práticas corolárias podem ser sugeridas. Em adição, para enriquecer mais ainda a avaliação, é interessante que as práticas sejam experimentadas em projetos em execução, e os efeitos documentados.

# REFERÊNCIAS


ANDERSON, D. J. **Kanban successful evolutionary change for technology organizations** – table of contents. [S.l.:s.n.], 2010. disponível em: <http:// www. scrumsense.com/wp-content/uploads/2010/02/KanbanManuscript-dja-rev1_072.pdf>. Acesso em: dez. 2014.

ANDRADE, M.; QUEIROZ, A.; QUEIROZ, R.. **Scrum-Seg: uma extensão da metodologia scrum para aplicações seguras.** Facol. Com, [S.l.], 2011. Disponível em: <http://www. facol.com/si/downloads/Revista_SI_2011/Artigo09.pdf>. Acesso em: dez. 2014.

ARKIN, B.; STENDER, S.; MCGRAW, G. Software penetration testing. **IEEE Security and Privacy**, v. 3, p. 84-7, 2005.

AYALEW, T.; KIDANE, T. **Identification and evaluation of security activities in agile projects** – a systematic literature review and survey study identification and evaluation of security activities in and agile. Karlskrona, set. 2012.

AZHAM, Z.; GHANI, I.; ITHNIN, N. **Security backlog in scrum security practices**. **MySEC**, [S.l.:s.n.], p. 414-17, 2011.

BACA, D.; CARLSSON, B. **Agile development with security engineering activities**. In: WORKSHOP ON SOFTWARE ENGINEERING FOR SENSOR NETWORK APPLICATIONS, 2., 2011. **Procedings**, [S.l.]: ACM City Editora, 2011. p. 149.

BARTSCH, S. Practitioners' perspectives on security in agile development. In: INTERNATIONAL CONFERENCE ON AVAILABITY, RELIABILITY, AND SECURITY, 6., 2011. **Procedings**, [S.l.:s.n.], 2011, p. 479-84.

BECK, K. **Extreme programming explained**: embrace change. [S.l.:s.n.], 2000.

BECK, K. et al. **Manifesto para o desenvolvimento ágil de software**. [S.l.:s.n.], 2001. Disponível em: <http://www.manifestoagil.com.br/>. Acesso em: 29 jan. 2015.

BERKELEY. **Incident response planning guideline**. Berkley: Berkley University, [200-]. Disponível em: <https://security.berkeley.edu/content/incident-response-planning-guideline>. Acesso em: 29 jan. 2015.

BEZNOSOV, K. **Extreme security engineering**: on employing XP practices to achieve Good Enough Security without defining it. [S.l.]: ACM, 2003.

BEZNOSOV, K.; KRUCHTEN, P. **Towards agile security assurance** – proceedings of the 2004 workshop on new security paradigms. New York: ACM, 2005.

BIG, T.; SMALL, A. 2013: Think Big, Act Small. The Standish Group International, CHAOS MANIFESTO, p. 1-52, 2013.

BOSTRÖM, G. et al. **Extending XP practices to support security requirements engineering**. In: INTERNATIONAL WORKSHOP ON SAFTWARE ENGINEERING FOR SECURE SYSTEMS, 1., 2006. **Anais…**, New York: ACM, 2006, p. 11–18.

COMMON CRITERIA. **About The Common Criteria**. Disponível em: <https://www.commoncriteriaportal.org/ccra/>. Acesso em: 29 jan. 2015.

CORREIA, M. P.; SOUSA, P. J. **Segurança no software**. Lisboa: FCA, 2010.

CHESS, B.; MCGRAW, G. Static analysis for security. **Security & Privacy**, [S.l.], v. 2, p. 76-9, 2004.

DE WIN, B. et al. On the secure software development process: CLASP, SDL and touchpoints compared. **Information and Software Technology**, [S.l.:s.n.], v. 51, p. 1152-71, 2009.

GIL, C. **Como elaborar projetos de pesquisa**. 4. ed. São Paulo: Atlas, 2002.

GE, X. et al. **Agile development of secure web applications**. In: INTERNATIONAL CONFERENCE ON WEB ENGINEERING, 6, 2006, New York. **Proceedings**, New York: ACM, 2006, p. 305-12.

GHANI, I.; YASIN, I. **Extreme programming methodology: a systematic literature**. Science International, [S.l.], v. 25, n. 2, p. 215-21, 2013.

GOOGLE**. Google Scholar**. Disponível em: <https://www.google.com.br/>. Acesso em: 05 fev. 2015.

GOOGLE. **Google**. Disponível em: <https://scholar.google.com.br>. Acesso em: 05 fev. 2015.

HIGHSMITH, J. **Accelerating enterprise agility adaptive leadership**. AGILE CONFERENCE, 1., 2011, Salt Lake City. Anais…, Salt Lake City: [S.n.], 2011.

IETF. **Internet Security Glossary**. Disponível em: <https://tools.ietf.org/html/rfc4949>. Acesso em: 29 jan. 2015.

KERAMATI, H.; MIRIAN-HOSSEINABADI, S.H. **Integrating software development security activities with agile methodologies**. INTERNATIONAL CONFERENCE ON COMPUTER SYSTEMS AND APPLICATIONS, 1. **Anais…**, 2008. p. 749-54.

MCGRAW, G.; PH, D. Software security touchpoint : architectural risk analysis cigital. **Technology**, [S.l.:s.n.], 2009.

MICROSOFT. **Boletins de Segurança**. [S.l.]: Microsoft Corporation. Disponível em: <https://technet.microsoft.com/pt-BR/library/security/dn631937.aspx>. Acesso em: 29 jan. 2015.

MICROSOFT. **Microsoft Security Lifecycle**. [S.l.]: Microsoft Corporation, [20--]. Disponível em: <http://www.microsoft.com/security/sdl/default.aspx>. Acesso em: 29 jan. 2015.

MICROSOFT. **SDL Agile**. [S.l.]: Microsoft Corporation, [20--]. Disponível em: <http://www.microsoft.com/security/sdl/discover/sdlagile.aspx>. Acesso em: 30 jan. 2015.

OFFENSIVE SECURITY. **Exploit Database**. Disponível em: <http://www.exploit-db.com/>. Acesso em: 05 fev. 2015.

MITRE CORPORATION. **CVE Details**: The ultimate security vulnerability datasource. Disponível em: <http://www.cvedetails.com/>. Acesso em: 05 fev. 2015.

OLIVEIRA, M. M. M. F.; EDS, J. M. P. **Lecture notes in computer science**. [S.l: s.n.].

OWASP. **CLASP**. 2014. Disponível em: <https://www.owasp.org/index.php/Category: OWASP_CLASP_Project>. Acesso em: 29 jan. 2015.

OWASP. **Evil user stories**, 2011. Disponível em: <https://www.owasp.org/index.php/ Agile_Software_Development:_Don't_Forget_EVIL_User_Stories>. Acesso em: 29 jan. 2015.

PEETERS, J. **Agile security requirements engineering**. **…** on Requirements Engineering for Information Security, 2005.

RUST, A.; BISHOP, B.; MCDAID, K. **Test-driven development:** can it work for spreadsheet engineering? [S.l: s.n.], 2006.

SATO, D. T. **Uso eficaz de métricas em métodos software Danilo Toshiaki Sato**. São Paulo: USP, 2007. Disponível em: http://www.teses.usp.br/teses/disponiveis/45/45134/ tde-06092007-225914/pt-br.php>. Acesso em: jan. 2015.

SAVOLA, R.; FRÜHWIRTH, C.; PIETIKÄINEN, A. Risk-driven security metrics in agile software development – an industrial pilot study. **Journal of Universal Computer Science**, [S.l.], v. 18, n. 12, p. 1679-702, 2012. Disponível em: http://www.jucs.org/jucs _18_12 /risk_driven_security_metrics>. Acesso em: dez. 2014.

SCHWABER, K.; SUTHERLAND, J. **The scrum guide** Scrum. org, October, v. 2, n. July, p. 17, 2011. <http://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf.>. Acesso em: dez. 2014.

SONIA; SINGHAL, A.; BANATI, H. FISA-XP: **an agile-based Integration of security activities with extreme programming**., New York, v. 39, n. 3, p. 1-14, 2014.

STALLINGS, W. **Criptografia e segurança de redes**: princípios e práticas. São Paulo: Pearson Pretice Hall, 2008.

STEVEN, E. J.; PETERSON, G. Cost-effective security. **Security & Privacy**, [S.l.], v. 5, n. 3, p. 64-6, 2007.

SOARES, M. **Comparação entre metodologias ágeis e tradicionais para o desenvolvimento de software**. Belo Horizonte: Unipac, 2006.

SOARES, F. **Sistemas de recompensa como estratégia de melhoria de produtividade em organizações de software**. VEÍCULOS, D. O. Universidade Federal de Pernambuco Centro de Informática Pós-Graduação em Ciência da Computação. 2009.

SOURCEFIRE. **Agile security manifesto:** effective security for the real world. [S.l.:s.n.]. Disponível em: <http://www.sourcefire.com/agile-security/manifesto>. Acesso em: 29 jan. 2015.

TAYLOR, D.; MCGRAW, G. Adopting a software security improvement program. **Security and Privacy**, [S.l.], v. 3, n. 3, p. 88-91, 2005.

TONIN, S. et al. **Uma análise de dívida técnica em uma empresa de tecnologia com desenvolvimento baseado em scrum**. In: BRAZILIAN WORKSHOP ON AGILE METHIODS LETTER FROM THE PROGRAM COMMITTEE, 3., 2012, São Paulo-SP. São Paulo: USP, 2012. p. 25-36.

VERSION one. **8º state of agile survey**. [S.l.:s.n.], 2013. Disponível em: <http://www.versionone.com/pdf/2013-state-of-agile-survey.pdf>. Acesso em: 29 jan. 2015.

WÄYRYNEN, J.; BODÉN, M.; BOSTRÖM, G. **Security Engineering and eXtreme Programming: An Impossible Marriage?** In: CONFERENCE ON EXTREME PROGRAMMING AND AGILE METHODS, 4., 2004, Calgary. **Proceedings...** Berlin: Springer, 2005. p. 117-28.

WAZLAWICK, R. S. **Metodologia de Pesquisa para Ciência da Computação**. Rio de Janeiro: Elsevier, 2008.

WILLIAMS, L.; MENEELY, A. **Protection poker : the new software security "game"**. [S.l.:s.n.], 2010.

WILLIAMS, L. **Agile software development methodologies and practices**: case study research: design and methods. 3. ed. Salt Lake City: Sage, 2003.

# APÊNDICE A – TABELAS UTILIZADAS PARA ANÁLISE COMPARATIVA

Quadro 1 – Resumo das estratégias de integração entre agilidade e segurança

(continua)

| REFERÊNCIA | SECURITY ACTIVITIES | TOOL | STRATEGY | FEEDBACK/ INTERVIEW/ VALIDATION |
|---|---|---|---|---|
| (SONIA; SINGHAL; BANATI, 2014) | CLASP | TISA-XP | Compara atividades de Segurança e atividades ágeis, verifica a viabilidade da integração delas. Utiliza métodos matemáticos e matrizes para tal. Envolve o cálculo do grau de agilidade das atividades, parâmetro de tolerância para redução das características ágeis. | Avaliado positivamente por uma empresa de desenvolvimento de software. |
| (KERAMATI, 2008) | Um estudo entre vários recursos acadêmicos e industriais selecionou 70 atividades. | - | Método para adição de segurança no processo ágil de desenvolvimento com um parâmetro ajustável para o controle das características ágeis do processo. Utiliza métodos matemáticos e matrizes para tal. Envolve o cálculo do grau de agilidade das atividades, parâmetro de tolerância para redução das características ágeis. | - |
| (BACA; CARLSSON, 2011) | Atividades extraídas do *Microsoft SDL*, *Cigatel Touchpoints* e *Common Criteria*. | - | Identificar quais práticas da Engenharia de Segurança podem ser integradas e beneficiar projetos ágeis e propor um processo ágil que utiliza atividades já estabelecidas nos processos de engenharia de segurança. | Entrevistados que trabalham com desenvolvimento de *software* apontam aspectos relevantes para a integração de engenharia de segurança com agilidade. |

(continuação)

| REFERÊNCIA | SECURITY ACTIVITIES | TOOL | STRATEGY | FEEDBACK/ INTERVIEW/ VALIDATION |
|---|---|---|---|---|
| (GHANI; YASIN, 2013) | Não foca em atividades específicas de segurança. | - | É feita revisão bibliográfica para responder/avaliar/classificar as seguintes questões:<br><br>Os elementos de segurança foram discutidos para XP e quantas pesquisas já mencionaram?<br><br>Quanto dos elementos de segurança mencionados na pesquisa podem ser uteis?<br><br>Existem modelos ou frameworks que relacionam segurança e XP?<br><br>Qual o nível de aceitação no ambiente de desenvolvimento de software a respeito do modelo XP? | - |
| (PEREIRA et al., 2011) | Não foca em atividades específicas de processos de segurança. | - | Ressalta a importância para a Segurança da Informação, da qualidade no desenvolvimento de sistemas informatizados, meta a ser atendida sem detrimento do desenvolvimento ágil. | - |

(continuação)

| REFERÊNCIA | SECURITY ACTIVITIES | TOOL | STRATEGY | FEEDBACK/ INTERVIEW/ VALIDATION |
|---|---|---|---|---|
| (AZHAM; GHANI 2011) | Trata de dos princípios de segurança e faz uma revisão bibliográfica sobre problemas de segurança atrelados ao Scrum. | - | Propõe um novo papel: "Security Master" e a criação do Security Backlog para tratar da gestão de segurança em projetos ágeis com Scrum. | - |
| (BARTSCH, 2011) | Não foca em atividades específicas de processos de segurança. | - | Expande o resultado de um estudo teórico, classificando os achados e aprimora o entendimento de segurança junto as práticas ágeis apresentando as perspectivas de profissionais de segurança no Desenvolvimento Ágil. | Entrevista com 10 entrevistados praticantes de agilidade de 9 empresas. As entrevistas tratavam de:<br>• Envolvimento do cliente;<br>• Experiência e percepção de segurança do desenvolvedor.<br>• Efeitos da agilidade na segurança.<br>• Práticas de segurança<br>• Autorização |

(continuação)

| REFERÊNCIA | SECURITY ACTIVITIES | TOOL | STRATEGY | FEEDBACK/ INTERVIEW/ VALIDATION |
|---|---|---|---|---|
| (BEZNOSOV, 2003) | Não foca em atividades específicas de processos de segurança. | - | Propõe aplicação de práticas do XP na engenharia de segurança e cita benefícios. | - |
| (WILLIAMS; MENEELY, 2010) | Não foca em atividades específicas de processos de segurança. | - | Motiva o uso do Protection Poker: útil para estimar riscos à segurança, adaptação da lista de requisitos à segurança, priorizar, incentivar debates com o time de desenvolvimento acarretando no compartilhamento de conhecimento sobre segurança de *software*. | Estudo de caso acompanhando um time de desenvolvimento. Resultado foi positivo. |
| (BOSTRÖM et al., 2006) | Atividades de segurança retiradas de padrões de segurança como o *Common Criteria*. | - | Propõe extensão do XP e suas práticas para oferecer suporte à engenharia de segurança e requisitos de segurança sem perder os benefícios das metodologias ágeis. Extensão do *Planning Game*. | Validação em um projeto para tese de mestrado. Em entrevistas estudantes afirmaram que era mais fácil de trabalhar com a extensão que com o *Common Criteria*. Porém dificuldades foram apontadas para a análise de risco. |
| (GE et al., 2006) | Não foca em atividades específicas de processos de segurança. | - | Processo ágil baseado no FDD com foco na análise de risco incremental e preparo de mecanismos de segurança a cada incremento e reavaliação das ameaças. | - |

(conclusão)

| REFERÊNCIA | SECURITY ACTIVITIES | TOOL | STRATEGY | FEEDBACK/ INTERVIEW/ VALIDATION |
|---|---|---|---|---|
| (BEZNOSOV; KRUCHTEN, 2004) | Não foca em atividades específicas de processos de segurança. | - | Analisa como quais práticas de segurança podem ser colocadas com processos ágeis. | - |
| (ANDRADE; QUEIROZ, 2011) | Não foca em atividades específicas de processos de segurança. | - | Extensão do Scrum preocupada com segurança da informação. Abordagem semelhante a do Azham mapeamento de elementos do *backlog* focados em segurança. | Em andamento |
| (PEETERS, 2008) | Não foca em atividades específicas de processos de segurança. | - | Extensão de *User Stories* utilizada em metodologias ágeis para a criação de *Abuse Stories*. | - |

Fonte: Arquivo da autora, 2015.

Quadro 2 – Pontos fortes e fracos das estratégias de integração

(continua)

| REFERÊNCIA | FORTES | PONTOS FRACOS |
|---|---|---|
| (SONIA; SINGHAL; BANATI, 2014) | • Validação na indústria de *software*<br>• Propõe Ferramenta.<br>• Similar à do Keremati.<br>• Proposta validada. | • Sem análise de custo. |

(continuação)

| REFERÊNCIA | FORTES | PONTOS FRACOS |
|---|---|---|
| (KERAMATI, 2008) | • Propõe método e algoritmo para a integração de segurança com práticas ágeis. | • Ausência de ferramenta suporte.<br>• Não apresenta validação/*feedback* da abordagem.<br>• Sem análise de custo. |
| (BACA; CARLSSON, 2011) | • Análise de vários processos de segurança.<br>• Abordagem teórica bem fundamentada.<br>• Entrevista. | • Ausência de ferramenta suporte. |
| (GHANI; YASIN, 2013) | • Forte embasamento teórico.<br>• Dados estatísticos. | • Ausência de ferramenta suporte.<br>• Sem análise de custo. |
| (PEREIRA et al., 2011) | • Aponta aspectos teóricos importantes para segurança sem detrimento de práticas ágeis. | • Segurança atrelada a metodologias ágeis é tratada superficialmente.<br>• Ausência de ferramenta suporte.<br>• Sem análise de custo.<br>• Não apresenta validação/ *feedback*/ entrevista. |
| (AZHAM; GHANI 2011) | • *Security Master*<br>• *Backlog Security* | • Ausência de ferramenta suporte.<br>• Sem análise de custo.<br>• Não apresenta validação/ *feedback*/ entrevista da abordagem. |

(continuação)

| REFERÊNCIA | FORTES | PONTOS FRACOS |
|---|---|---|
| (BARTSCH, 2011) | • Classifica os desafios apontados pela bibliografia.<br>• Classifica as estratégias de mitigação apontadas pela bibliografia.<br>• Entrevistas<br>• Abordagem prática. | • Entrevistas subjetivas e realizadas com apenas 10 profissionais praticantes de desenvolvimento de software com práticas ágeis. Ou seja, amostra pouco representativa.<br>• Sem análise de custo.<br>• Ausência de ferramenta suporte. |
| (BEZNOSOV, 2003) | • Apresenta abordagem prática.<br>• XSE: Integra de fato práticas ágeis com engenharia de segurança. | • Não beneficia qualquer projetos de engenharia de segurança.<br>• Ausência de ferramenta suporte.<br>• Sem análise de custo.<br>• Não apresenta validação /*feedback*/ entrevista da abordagem. |
| (WILLIAMS; MENEELY, 2010) | • Abordagem prática<br>• Estudo de caso<br>• Dados estatísticos importantes. | • Estudo de caso com amostra pouco representativa.<br>• Ausência de ferramenta suporte. |

(continuação)

| REFERÊNCIA | FORTES | PONTOS FRACOS |
|---|---|---|
| (BOSTRÖM et al., 2006) | • Abordagem prática<br>• Validação<br>• Mais fácil de utilizar do que padrões já estabelecido.<br>• Preocupado com a agilidade do processo. | • Ausência de ferramenta suporte.<br>• O feedback relatou dificuldade para análise de risco. |
| (GE et al., 2006) | • Foco nas aplicações *web*<br>• Detalhamento da análise de risco à segurança.<br>• Abordagem prática | • Ausência de ferramenta suporte.<br>• Não apresenta validação /*feedback*/ entrevista da abordagem.<br>• Sem análise de custo. |
| (WÄYRYNEN; BODÉN; BOSTRÖM, 2005) | • Abordagem teórica bem fundamentada.<br>• Proposta do XP integrado com padrões de segurança. | • Sem análise de custo.<br>• Ausência de ferramenta suporte.<br>• Não apresenta validação /*feedback*/ entrevista da abordagem.<br>• Não aplicável para sistemas de segurança críticos / tempo real. |
| (BEZNOSOV; KRUCHTEN, 2004) | • Classifica técnicas e métodos de segurança e sua integração com o desenvolvimento ágil. | • Sem análise de custo.<br>• Ausência de ferramenta suporte.<br>• Não apresenta validação /*feedback*/ entrevista da abordagem. |

(conclusão)

| REFERÊNCIA | FORTES | PONTOS FRACOS |
|---|---|---|
| (ANDRADE; QUEIROZ, 2011) | • *Security Master*<br>• *Backlog Security* | • Sem análise de custo.<br>• Ausência de ferramenta suporte.<br>• Não apresenta validação /*feedback*/ entrevista da abordagem. |
| (PEETERS, 2008) | • Flexibilidade à mudanças de requisitos e revisão de aspectos de segurança. | • Falta de rastreabilidade<br>• Ausência de ferramenta suporte.<br>• A documentação reduzida dificulta a identificação de como o sistema reage em cada *Abuse Case*.<br>• Não apresenta validação /*feedback*/ entrevista da abordagem. |

Fonte: Arquivo da autora, 2015.