PASS: Processo de Apoio à Segurança de Software

Francisco José Barreto Nunes[1], Arnaldo Dias Belchior (In Memorium)

Mestrado em Informática Aplicada - Universidade de Fortaleza (UNIFOR)
Av. Washington Soares, 1321 CEP 60.811-341, Fortaleza – CE – Brasil

[1]fcojbn@yahoo.com.br

Abstract. *The need of developing software products with increased quality demanded the creation of standards and maturity models related to the quality of development process and software products. However, despite the investment made in software processes development, there is still no assurance that the developed systems are immune to attacks or do not present security problems. This paper proposes a formal process aimed at improving software security, the Process to Support Software Security (PSSS), which was based on the result of a research field and validated by a case study.*

Resumo. *A necessidade de desenvolver produtos de software com maior qualidade exigiu a criação de normas e modelos de maturidade voltados para a qualidade do processo de desenvolvimento e de produtos de software. Contudo, apesar dos recursos investidos em processos de desenvolvimento de software ainda não há garantia de que os sistemas desenvolvidos sejam imunes a ataques ou deixem de apresentar problemas de segurança. Este trabalho propõe um processo formal orientado a elevar a segurança de software, o Processo de Apoio à Segurança de Software (PASS), que foi baseado no resultado de uma pesquisa de campo e validado por meio de um projeto piloto.*

1. Introdução

Os modelos de maturidade, como o CMMI (2006), e as normas internacionais orientadas à melhoria de processos de desenvolvimento de software, como a ISO/IEC 12207 (2002), ainda não enfatizam a capacidade dos sistemas de resistir a ataques e manter a segurança das informações manipuladas.

A disciplina Segurança da Informação almeja proteger as informações manipuladas pelos sistemas de informação. Essa meta se ratifica por meio da correlação da segurança com processos de negócio, incluindo, nesse contexto, os processos de desenvolvimento de software. Dias (2000) afirma que a Segurança da Informação é muito ampla e que seus objetivos envolvem a manutenção da confidencialidade, a disponibilidade e a integridade de informações, variando de acordo com o tipo de ambiente computacional e com o grau de importância do sistema.

McGraw (2006) comenta que o aumento da complexidade e da interconectividade dos sistemas, além da necessidade do mercado de receber rapidamente as soluções de software (*time to market*) contribuem para um número cada vez maior de sistemas com falhas de segurança.

Outro fator é que os processos de desenvolvimento de software são estruturados sem considerar a segurança do sistema desenvolvido. Segundo McGraw (2004), a

segurança de software não pode ser confundida com software seguro. Isto é, funcionalidades e características de segurança presentes em um software não o tornam seguro.

Este trabalho objetiva apresentar o Processo de Apoio à Segurança de Software (PASS) que contribui para a construção de software seguro e, por conseguinte, software de maior qualidade. O PASS foi estruturado a partir do SSE-CMM (2003), do OCTAVE [ALBERTS *et al*. 2001], da ISO/IEC 15408 (2005a, 2005b, 2005c), e da ISO/IEC 27002 (2005), organizado por meio de pesquisa de campo e experimentado por meio da aplicação em projeto piloto.

Este trabalho está organizado da seguinte forma: a seção 2 apresenta brevemente as abordagens de segurança utilizadas para estruturar as atividades do processo de apoio proposto. A seção 3 comenta o principal resultado obtido com a pesquisa de campo. A seção 4 descreve o Processo de Apoio à Segurança de Software (PASS). A seção 5 discorre sobre a experiência prática de aplicação do PASS. A seção 6 apresenta as principais conclusões deste trabalho.

2. Abordagens de Segurança da Informação

2.1 SSE-CMM

O *International Systems Security Engineering Association* [ISSEA 2003] estendeu o CMM (*Capability Maturity Model*) [PAULK *et al*. 1993] para o SSE-CMM (*System Security Engineering – Capability Maturity Model*) (2003). A versão anterior do SSE-CMM tornou-se a norma ISO/IEC 21827 (2002).

O SSE-CMM propõe 22 áreas de processo utilizadas para obter a segurança na tecnologia da informação. O escopo abrangente do modelo torna-o difícil de aplicar e demanda muitos recursos. Para a proposta do PASS, apenas as áreas de processo relacionadas com engenharia de segurança foram consideradas neste trabalho, uma vez que o objetivo é propor um processo seguro de desenvolvimento de software.

2.2 OCTAVE

O OCTAVE (*Operationally Critical Threat, Asset, and Vulnerability Evaluation*) [ALBERTS *et al*. 2001] é uma técnica auto-orientada de avaliação de riscos para segurança. Isto é, os gestores e executores dos processos organizacionais são os responsáveis por configurar a estratégia de risco da própria organização, uma vez que são aqueles que melhor conhecem os problemas e vulnerabilidades desses processos.

As variáveis relacionadas com os riscos (ativos, ameaças, vulnerabilidades, e impacto organizacional) são consideradas nas tomadas de decisão, permitindo que uma organização ajuste a estratégia de proteção de seus riscos de segurança.

Os requisitos do OCTAVE são agrupados em um conjunto de critérios. Assim sendo, dois métodos consistentes com os critérios foram desenvolvidos: o OCTAVE e OCTAVE-S [ALBERTS *et al.* 2003]. O OCTAVE foi projetado para ser aplicado em grandes organizações. O OCTAVE-S é voltado para pequenas organizações. O processo proposto baseia-se tanto no OCTAVE quanto no OCTAVE-S.

2.3 ISO/IEC 15408

A ISO/IEC 15408 (*Evaluation Criteria for Information Technology Security*) (2005a, 2005b, 2005c) propõe um conjunto de critérios para avaliar a segurança de produtos de tecnologia da informação. A ISO/IEC 15408 deriva-se do *Common Criteria* (2005).

O desenvolvimento seguro segundo a ISO/IEC 15408 se baseia na realização de atividades que representam uma seqüência a partir da qual os requisitos e as especificações de segurança são derivados.

A ISO/IEC 15408 considera 2 tipos de requisitos de segurança. Os requisitos funcionais representam um conjunto mínimo de características de segurança a serem embutidas em um software. Os requisitos de garantia representam atividades de apoio ao desenvolvimento de forma a acreditar que o software seja seguro por ter realizado tais atividades.

2.4 ISO/IEC 27002

Segundo a ISO/IEC 27002 (2005), a informação é um ativo de valor para a organização e, conseqüentemente, necessita preservar sua confidencialidade, integridade e disponibilidade por meio da implantação de controles.

Os controles podem incluir políticas, práticas ou processos organizacionais os quais devem garantir que os objetivos estabelecidos para a segurança foram satisfatoriamente atendidos.

3. Importância das Atividades de Segurança no Processo de Apoio

Com as abordagens apresentadas na seção anterior, tendo como principal referência o SSE-CMM (2003), realizou-se um mapeamento no contexto de um processo de desenvolvimento de software, envolvendo planejamento, análise, e passando pela verificação e validação, e garantia da qualidade.

A partir desse mapeamento, um processo inicial composto de atividades de segurança foi estruturado. Esse processo foi organizado a partir do resultado de uma pesquisa de campo e, em seguida, recebeu sua denominação final como Processo de Apoio à Segurança de Software (PASS).

O conjunto inicial de atividades de segurança propostas para compor o PASS foi avaliado por meio da pesquisa de campo com a participação de 42 especialistas em processo de desenvolvimento de software e especialistas em segurança da informação de várias instituições públicas e privadas distribuídas entre as regiões do Brasil.

Com os resultados obtidos nessa pesquisa de campo, algumas atividades avaliadas foram renomeadas para melhor compreensão, culminando com as atividades do Processo de Apoio à Segurança de Software. A condução desta pesquisa de campo ocorreu a partir da aplicação de um questionário baseado em Andrade (2005), em que foi identificado o grau de importância das 40 atividades inicialmente propostas para o processo.

A Figura 1 mostra, em ordem decrescente do grau de importância, as sete atividades melhor avaliadas do processo seguro proposto, por todos os especialistas desta pesquisa. Comparando-se os resultados das avaliações realizadas por especialistas de processo de desenvolvimento de software e especialistas em segurança, é percebido que cinco das sete atividades melhor pontuadas são as mesmas tanto na avaliação dos

especialistas em processo de software quanto na avaliação dos especialistas em segurança da informação, destacado na Figura 1.

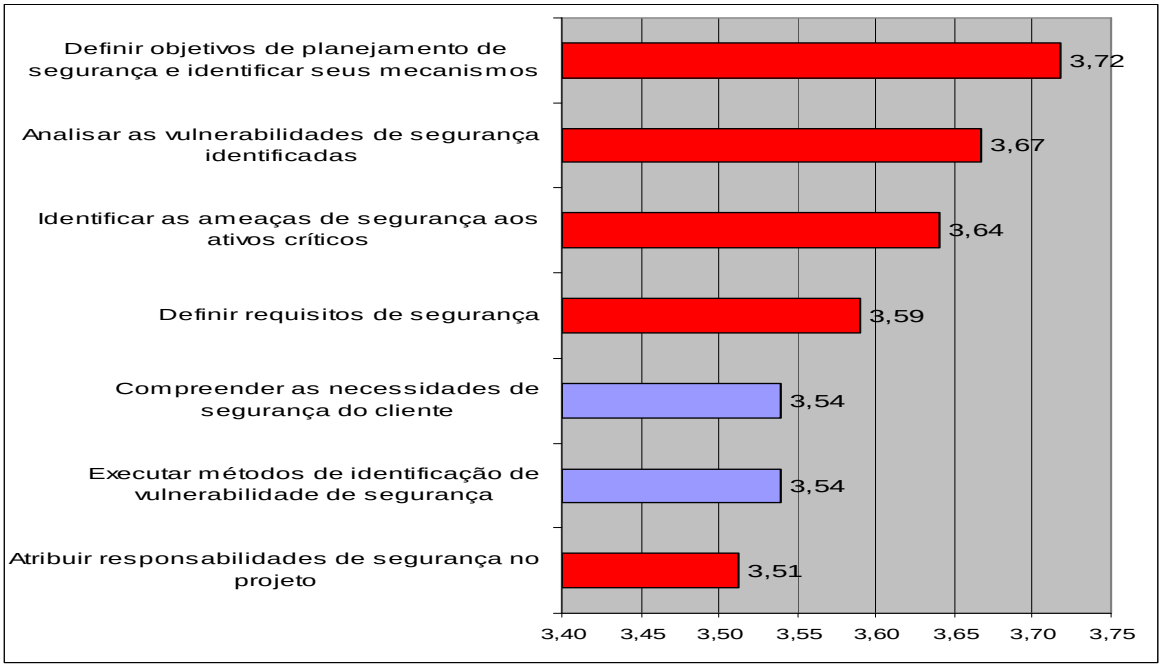

Figura 1. Atividades melhor avaliadas no processo de apoio

4. Processo de Apoio à Segurança de Software

O PASS (Figura 2) foi elaborado inicialmente tendo como base o ciclo de vida iterativo incremental. Este processo é constituído de 37 atividades agrupadas em 11 subprocessos de segurança. Cada atividade é descrita em termos de propósito, tarefas, artefatos de entrada, artefatos de saída, e atores envolvidos.

O PASS está estruturado de modo a atender, principalmente, aos seguintes objetivos: (1) Dar mais visibilidade para a segurança de software em um processo de desenvolvimento de software; (2) Tratar sistematicamente vulnerabilidades, ameaças, impactos e riscos de segurança relacionados ao produto de software; e (3) Possibilitar que ações de segurança estejam alinhadas com os objetivos estratégicos especificados para a organização.

A Figura 2 representa também os subprocessos através de cores que, conforme a legenda, são propostas genéricas que tentam associar os subprocessos com algumas disciplinas do RUP (2003). Essa representação é uma sugestão de trabalho futuro cuja idéia macro é identificar onde o PASS poderia contribuir com a inclusão de novas atividades nas disciplinas ou fases do RUP.

Os subprocessos de segurança elencados representam uma proposta não exaustiva. Neste contexto, é recomendável especializar o processo proposto para a organização que o utilizar, segundo suas necessidades e peculiaridades. Ou seja, é possível utilizar apenas um subconjunto dos subprocessos e ou atividades propostos, adaptá-los e ou incluir novas atividades específicas. A partir do processo de apoio especializado para a organização, poderá haver então sua instanciação para a execução dos projetos de software. Na instanciação do processo de apoio para os projetos, nem todos os artefatos podem ser requeridos ou gerados em cada atividade.

O PASS propõe dois papéis que apóiam a execução do processo: o *Engenheiro de segurança*, e o *Auditor de segurança*. O Engenheiro de segurança especializa e instancia o processo de apoio para os objetivos do projeto, alinhado com metas e planos da organização, e monitora se esses projetos satisfazem os objetivos de segurança. O *Auditor de segurança* é responsável pela avaliação da aderência do processo de apoio nos projetos de software, em termos de resultados das atividades, dos artefatos produzidos, verificando a conformidade das ações dos projetos ao processo estabelecido.

Os 11 subprocessos do PASS serão resumidamente descritos a seguir.

4.1 Planejar segurança

O propósito é garantir que as informações necessárias para o planejamento da segurança para o projeto sejam estabelecidas e registradas.

Devem ser elaborados (ou refinados) os objetivos e o plano de segurança da informação para a organização, e ser constituída a equipe de segurança (quando da instanciação do processo de apoio).

O engenheiro de segurança é responsável por definir junto à equipe do projeto qual a visão de segurança a ser estabelecida para o projeto. Atua também como consultor de segurança de software para essa equipe.

Este subprocesso é composto pelas seguintes atividades: (i) Desenvolver Plano de segurança; (ii) Planejar ambientes de processamento; e (iii) Planejar o gerenciamento de incidentes de segurança.

4.2 Avaliar vulnerabilidade de segurança

O propósito é identificar e caracterizar (na iteração) vulnerabilidades de segurança do software relacionadas com um ambiente definido, normalmente no ambiente onde o sistema será utilizado.

O engenheiro de segurança é responsável por conduzir os métodos de identificação de vulnerabilidades de segurança e analisá-las, a partir de discussões conjuntas com os principais usuários - membros dos níveis estratégico (executivos), tático (gerência), e operacional (técnico) - e com o apoio do engenheiro de software.

Este subprocesso é composto pelas seguintes atividades: (i) Identificar vulnerabilidades de segurança; e (ii) Analisar as vulnerabilidades de segurança identificadas.

4.3 Modelar ameaça de segurança

O propósito é identificar e caracterizar ameaças de segurança com suas propriedades e características, a partir das vulnerabilidades resultantes do subprocesso anterior.

O engenheiro de segurança é responsável por conduzir a identificação das ameaças de segurança, na iteração, elaborando casos de abuso e árvores de ataque, classificá-las, e desenvolver estratégias de redução das ameaças, a partir de discussões conjuntas com os principais usuários - membros dos níveis estratégico (executivos), tático (gerência), e operacional (técnico) - e apoio do engenheiro de software.

Este subprocesso é composto pelas seguintes atividades: (i) Identificar as ameaças de segurança; (ii) Classificar as ameaças de segurança; e (iii) Desenvolver estratégias de redução das ameaças de segurança.

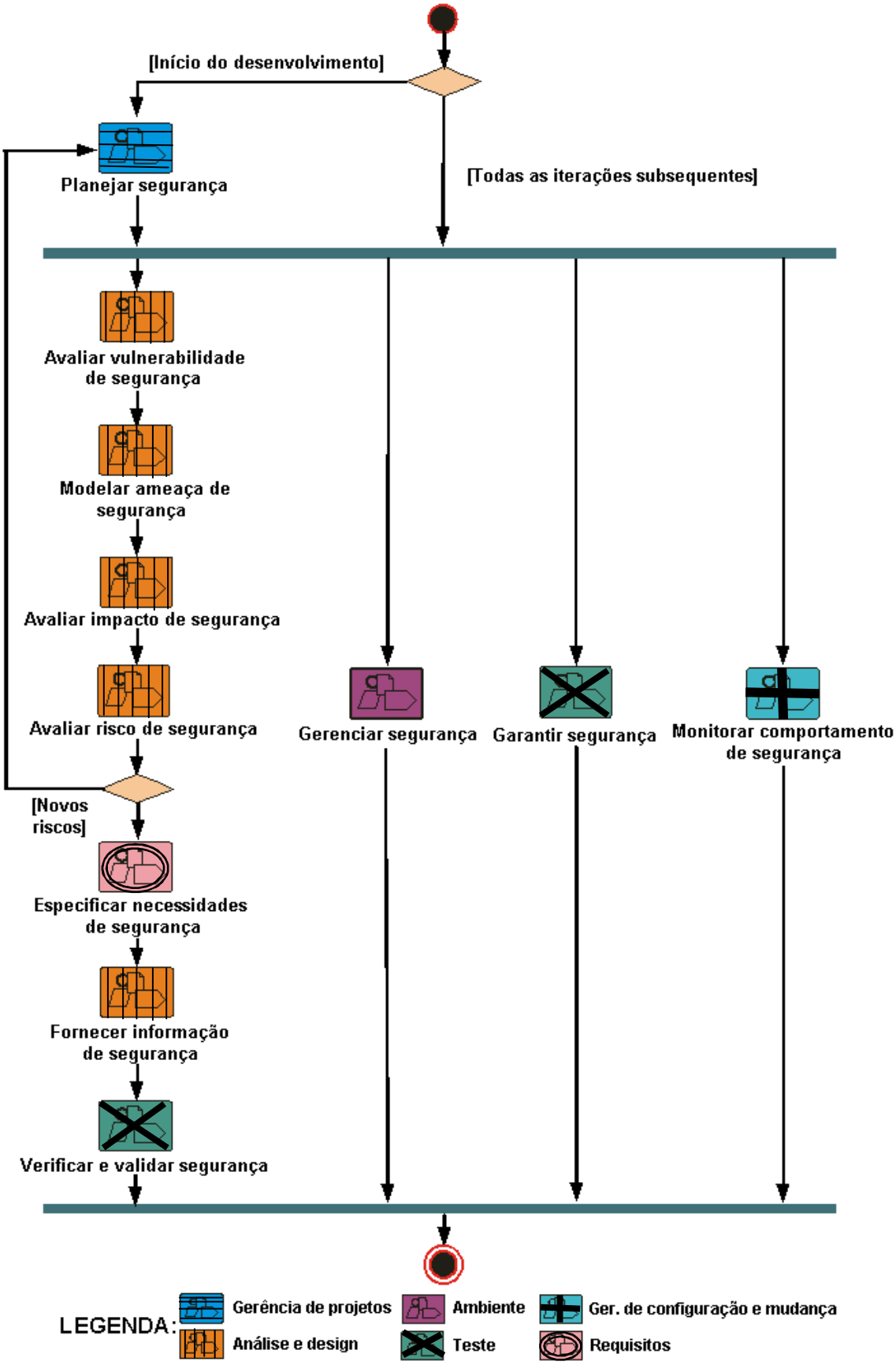

Figura 2. Processo de apoio à segurança de software

4.4 Avaliar impacto de segurança

O propósito é identificar e caracterizar impactos que sejam relevantes com respeito ao software e avaliar a possibilidade da ocorrência desses impactos.

O engenheiro de segurança, engenheiro de software, e o usuário (cliente) são responsáveis pela priorização das atividades críticas influenciadas pelo software. O engenheiro de segurança e o engenheiro de software tratam da revisão dos artefatos de software que se referem à segurança. A partir das informações anteriores e pela avaliação de cada uma das ameaças e das vulnerabilidades, o engenheiro de segurança, sendo apoiado pelo usuário, identifica os impactos de segurança de incidentes não desejados e descreve informações detalhadas sobre estes, podendo cobrir os custos diretos e em cascata envolvidos por causa dos diferentes riscos de desenvolvimento do produto ou da iteração.

Este subprocesso é composto pelas seguintes atividades: (i) Tratar as atividades críticas para segurança; (ii) Revisar artefatos do software que impactam na segurança; e (iii) Identificar e descrever impactos de segurança.

4.5 Avaliar risco de segurança

Com as informações de vulnerabilidade, ameaça e impacto de segurança, o propósito é avaliar os riscos inerentes do software em desenvolvimento pela identificação da exposição de segurança, o risco dessa exposição, e a priorização desses riscos. A Figura 3 exemplifica o diagrama que representa o subprocesso “Avaliar Risco de Segurança”.

O engenheiro de segurança, com apoio do usuário, é responsável pela identificação da exposição de segurança, pela avaliação do risco da exposição, e pela priorização desses riscos. O engenheiro de software pode também auxiliar a definir a priorização dos riscos de segurança.

Este subprocesso é composto pelas seguintes atividades: (i) Identificar exposição de segurança; (ii) Avaliar risco de exposição de segurança; e (iii) Priorizar riscos de segurança.

4.6 Especificar necessidades de segurança

O propósito é especificar as necessidades relacionadas com segurança para o software entre todos os envolvidos (principalmente o usuário), em cada iteração.

O engenheiro de segurança, com apoio do usuário e do cliente, é responsável pela compreensão das necessidades de segurança do cliente, desenvolvendo uma visão de alto nível orientada à segurança do software. A visão de alto nível ajuda na definição dos requisitos de segurança. Finalmente, o engenheiro de software faz a mediação para obter acordo sobre esses requisitos de segurança.

Este subprocesso é composto pelas seguintes atividades: (i) Compreender as necessidades de segurança do cliente; (ii) Capturar uma visão de alto nível orientada à segurança do software; (iii) Definir requisitos de segurança; e (iv) Obter acordo sobre requisitos de segurança.

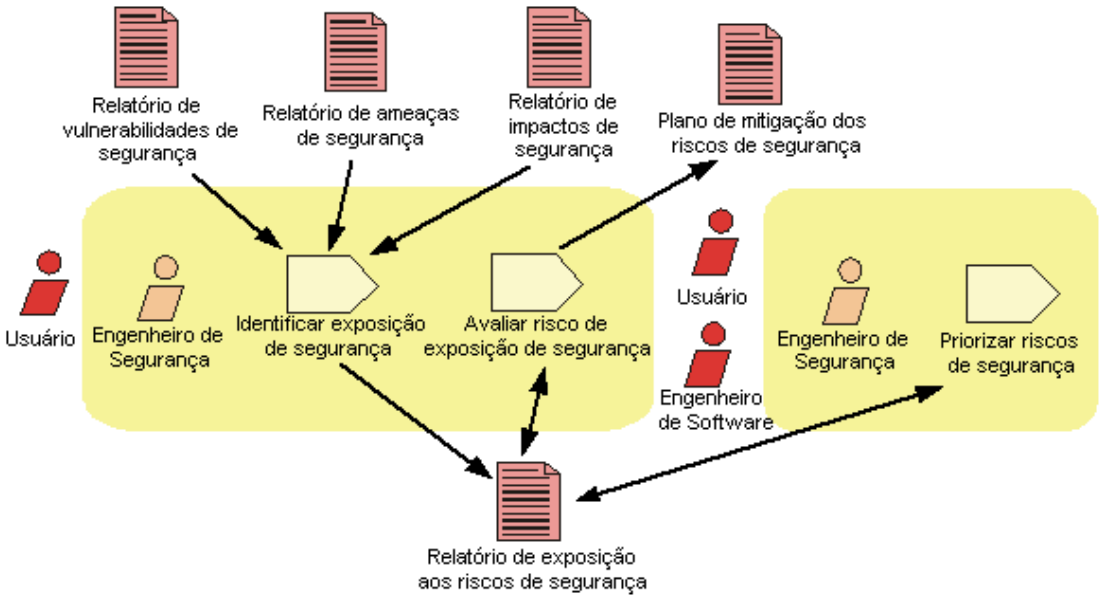

Figura 3. Subprocesso: Avaliar risco de segurança

4.7 Fornecer informação de segurança

O propósito é prover aos arquitetos de software, projetistas, analistas, programadores, ou usuários informações de segurança de que eles necessitam.

Este subprocesso procura gerar informação a fim de que: o máximo de problemas do software seja revisado em relação a implicações de segurança e seja resolvido de acordo com objetivos de segurança; os membros da equipe de projeto compreendam o processo de apoio e entendam aspectos da segurança da informação, para que possam realizar suas funções; e a solução reflita a informação de segurança fornecida. O Engenheiro de Segurança atua como "*mentoring*" de segurança para os demais componentes da equipe de projeto.

Este subprocesso é composto pelas seguintes atividades: (i) Entender necessidades de informação de segurança; (ii) Identificar restrições e ponderações de segurança; (iii) Fornecer requisitos de apoio ao processo; e (iv) Fornecer alternativas de segurança.

4.8 Verificar e validar segurança

O propósito é garantir que as soluções sejam verificadas e validadas em relação à segurança ao longo do processo de desenvolvimento.

O engenheiro de segurança, sendo auxiliado pelo engenheiro de software, define a abordagem de verificação e validação de segurança a qual envolve a elaboração de planos, a cobertura e a profundidade dos testes de verificação e validação. Em seguida, o engenheiro de segurança, com o apoio do SQA (garantia da qualidade), realiza a verificação e validação de segurança, revisando ao final os resultados e comunicando-os. O auditor de segurança acompanha a realização da verificação e da validação a fim de averiguar a correta realização das atividades do subprocesso.

Este subprocesso é composto pelas seguintes atividades: (i) Definir a abordagem de verificação e validação de segurança; (ii) Realizar verificação de segurança; (iii)

Realizar validação de segurança; e (iv) Revisar e comunicar resultados de verificação e validação de segurança.

4.9 Gerenciar segurança

O propósito é tratar das tarefas necessárias para administrar e manter os mecanismos de controle de segurança para o ambiente de desenvolvimento, bem como gerenciar a implantação de controles para as funcionalidades de segurança, que se integram com os controles existentes.

Define também como será o gerenciamento da segurança, envolvendo a definição dos treinamentos e programas de educação de segurança necessários. Os serviços de segurança e mecanismos de controle para o projeto de desenvolvimento específico são detalhados. Outra atribuição é resolver os problemas identificados com a verificação e validação de segurança.

O engenheiro de segurança gerencia e controla serviços de apoio e componentes operacionais de segurança, identificando necessidades de treinamento e realizando ações de conscientização sobre o processo de apoio para desenvolver software mais seguro e sobre assuntos de segurança da informação. Gerencia também a implantação de controles de segurança nos produtos de software. O gerente de projeto auxilia o engenheiro de segurança no gerenciamento destas atividades.

Este subprocesso é composto pelas seguintes atividades: (i) Gerenciar serviços e componentes de segurança; (ii) Gerenciar treinamento e programas de educação de segurança; e (iii) Gerenciar a implementação de controles de segurança.

4.10 Garantir segurança

O propósito é definir um conjunto de atividades que possam ser aplicadas para garantir que a segurança do produto foi alcançada e será mantida.

O engenheiro de segurança e o engenheiro de software são responsáveis pelo desenvolvimento de uma estratégia de manutenção da garantia de segurança, sendo auxiliados pelo cliente e pelo auditor de segurança. O auditor de segurança é responsável pela condução da análise de impacto das mudanças em relação à segurança para atestar que nenhuma mudança comprometa a segurança do software. O engenheiro de software e o auditor de segurança controlam as evidências da garantia de segurança, que ratificam esta manutenção.

Este subprocesso é composto pelas seguintes atividades: (i) Definir estratégia de garantia de segurança; (ii) Conduzir análise de impacto de segurança das mudanças; e (iii) Controlar as evidências da garantia de segurança.

4.11 Monitorar comportamento de segurança

O propósito é monitorar a segurança do software ou de suas partes liberadas ao longo de sua iteração em seu estado operacional. Isto assegura que deficiências ou erros que poderiam conduzir a uma falha de segurança sejam monitorados e, por conseguinte, identificados, reportados e acompanhados até suas correções.

O engenheiro de segurança, apoiado pelo engenheiro de software, é responsável pelas atividades de monitoração do comportamento de segurança. O auditor de segurança acompanha essas atividades com intuito de identificar possíveis não

conformidades, reavaliando mudanças nas ameaças, vulnerabilidades, impactos, riscos e ambiente.

Este subprocesso é composto pelas seguintes atividades: (i) Analisar registro de evento com impacto na segurança; (ii) Identificar e preparar a resposta dos incidentes de segurança relevantes; (iii) Monitorar mudanças em ameaças, vulnerabilidades, impactos, riscos, e no ambiente; (iv) Revisar o comportamento de segurança do software para identificar mudanças necessárias; e (v) Realizar auditorias de segurança.

Na seção seguinte, será apresentada a experiência prática da aplicação do PASS.

5. Aplicação do PASS

O Processo de Apoio à Segurança de Software foi aplicado em uma organização da administração pública no Ceará com mais de 4 milhões de clientes, cuja tecnologia é uma atividade meio, e cuja equipe é formada por 30 analistas e programadores. A metodologia de desenvolvimento de software da empresa é baseada no PMBOK (2004) e no RUP (2003). Salienta-se que a empresa, até então, não possuía processo relacionado à segurança de software.

O PASS foi aplicado em projeto de sistema de auditoria e controle de acesso considerado crítico pela empresa. Ou seja, um incidente de segurança no sistema causaria um impacto negativo à organização. Essa associação com a criticidade facilitou aceitação do PASS pelo patrocinador do projeto e demais gestores. Contudo, destaca-se que o PASS foi estruturado para ser aplicável no desenvolvimento de todos os tipos de produtos de software.

O sistema da experiência, que possui arquitetura *Web*, controla a segurança de acesso lógico de todos os usuários dos sistemas corporativos dessa organização, por meio da associação de perfis para cada tipo de usuário. Ele também padroniza a criação, alteração e manutenção de senhas, além de configurar e registrar dados para auditoria. Todas as funcionalidades do sistema foram formalizadas em nove casos de uso.

Devido às restrições de prazo de 4 semanas, não foi possível especializar o PASS de forma adequada para utilização no projeto. Logo, a experência prática de aplicação do PASS contemplou apenas as atividades melhor avaliadas por todos os especialistas participantes, conforme Figura 1.

A equipe era formada por 4 colaboradores, sendo 1 gestor de projeto e 2 analistas programadores dedicados ao projeto, além da figura do engenheiro de segurança.

Inicialmente, foi apresentada e explicada cada uma das atividades ao gerente de projeto e aos analistas responsáveis. Avaliou-se conjuntamente a aplicabilidade das atividades selecionadas à realidade do projeto com intuito de melhor ajustá-las à metodologia de desenvolvimento corporativa, reduzindo impactos negativos. Ao término, identificaram-se os objetivos de segurança almejados para o sistema.

Em seguida, foram relatados para o patrocinador, os gestores das áreas de negócio, e o gestor de informática os benefícios e a importância de se aplicar o processo de apoio. Dessa forma, avaliou-se a receptividade da alta administração, e a aplicabilidade do processo à realidade do projeto.

Os estágios iniciais da aplicação do processo de apoio apresentaram as seguintes peculiaridades:

- Ajustes nas atividades selecionadas a fim de melhorar sua aplicação no projeto.
- Necessidade de uma prática para identificar os principais ativos de informação do software.
- Necessidade de uma base histórica de erros e problemas nos sistemas para ajudar na identificação de possíveis vulnerabilidades e ameaças de segurança.
- Necessidade de mais tempo para avaliar as vulnerabilidades e modelar as ameaças.

Após o término da análise de perigo e a identificação de alguns problemas e suas vulnerabilidades, tais vulnerabilidades foram detalhadas e encaminhadas aos analistas e ao principal usuário para análise adicional e revisão final. A Tabela 1 apresenta um resumo da lista de vulnerabilidades identificadas.

Tabela 1. Exemplo de vulnerabilidades de segurança

Vulnerabilidade	Descrição	Criticidade
04. Falha na função de validação da senha	Permitir que usuários criem senhas de fácil descoberta comprometerá a segurança do acesso aos sistemas	Alta
08. Falha de segurança na manutenção dos registros de auditoria	As bases e registros de auditoria que não sejam protegidos podem sofrer alterações indevidas para acobertar ações ilegais que ocorram na operação de algum sistema.	Alta

A utilização de árvores de ataque (Figura 4) auxiliou na identificação das ameaças de segurança aos ativos (artefatos) críticos do sistema e na definição da capacidade das ameaças ocorrerem. Como última atividade, a relação de ameaças foi aprovada.

O entendimento das necessidades de segurança ocorreu com o apoio das informações sobre as vulnerabilidades e ameaças de segurança. A partir da definição dessas necessidades e com apoio de casos de abuso e árvores de ataque, os requisitos de segurança foram organizados. Os requisitos de segurança estão representados no Quadro 1.

Depois do aceite formal dos requisitos por parte do patrocinador e principais usuários, o subprocesso de verificar e validar a segurança foi iniciado. Devido ao curto prazo para a entrega do sistema, somente inspeções nos códigos foram conduzidas. Após realização das verificações de segurança, a aplicação do PASS foi considerada concluída.

Apesar da obtenção dos requisitos de segurança, do término do projeto no prazo, e da boa receptividade por parte dos gestores, ocorreram alguns problemas, sendo que os principais incluem:

- Demora em entender e assimilar a aplicação do processo de apoio, demandando maior tempo para a execução do projeto piloto.
- Dificuldade em alinhar as atividades do processo de apoio com as atividades da metodologia de desenvolvimento de sistemas da organização.

Figura 4. Exemplo de árvore de ataque do sistema de auditoria e segurança

- Necessidade de recursos adicionais e de ajustes para aumentar a eficácia da aplicação do processo de apoio.

As principais vantagens obtidas ao se aplicar e seguir o PASS incluem:

- Garantia de que a segurança foi considerada durante o desenvolvimento do sistema e que as vulnerabilidades, ameaças, e riscos potenciais de segurança foram tratados.
- Identificação e definição dos requisitos de segurança baseados em um conjunto de avaliações de forma a proteger o sistema contra problemas de segurança.
- Garantia de que os recursos limitados do projeto foram aplicados eficazmente baseados em avaliações de segurança e de acordo com os maiores impactos de segurança.

Quadro 1. Requisitos de segurança (RS) para o sistema de auditoria e segurança

Identificação	**Descrição**
RSS.01	**Impedir criação de senhas inseguras**
Toda e qualquer criação ou alteração de senha, seja realizada pelo sistema ou pelo usuário, deve seguir, no mínimo, as regras que estabelecem a norma interna de elaboração de senhas. Sempre e quando o usuário compor senhas que não respeitem tais regras, o sistema não permitirá sua criação, solicitando ao usuário a elaboração de outra senha. A criação só será efetivada quando respeitar as regras impostas.	
RSS.02	**Impedir acesso indevido ao sistema**
Os acessos do sistema devem ser seguros com autenticação e validação de perfil a fim de evitar: (*i*) Envio incorreto da senha inicial gerada pelo sistema, (*ii*) Pessoas estranhas acessem algum sistema se passando por usuários válidos, (*iii*) Roubo, alteração ou qualquer outra ação que comprometa as informações dos usuários, (*iv*) O usuário do sistema acessa outras informações além daquelas atribuídas ao seu grupo, (*v*) Cadastro de informações incorretas. Revisões periódicas dos usuários e seus grupos/perfis poderiam reduzir acessos indevidos.	

Identificação	Descrição
RSS.03	**Impedir alteração nos registros de auditoria**
Os registros (log) das ações realizadas pelos usuários e pelo administrador do sistema serão armazenados para consulta e estes registros não devem ser alterados ou apagados por estes usuários ou pelo administrador.	

6. Conclusão

Uma vez que somos responsáveis pela institucionalização da qualidade de software, deveríamos perceber que a segurança de software não é mais um assunto desconhecido ou insignificante a ponto de não considerá-lo em projetos de desenvolvimento de software.

Sobremaneira, somos todos co-responsáveis por garantir a segurança dos sistemas que ajudamos a desenvolver. Muitos especialistas em qualidade de software podem até compreender a importância de projetar para segurança, mas poucos conhecem como aplicá-la no ciclo de vida de desenvolvimento [MCGRAW 1999].

Segundo Howard (2002), a segurança de software deveria ser organizada e incorporada desde o início do ciclo de vida do software. Só assim é que se obteriam sistemas mais seguros.

O Processo de Apoio à Segurança de Software (PASS) é formado por atividades de segurança a serem incorporadas durante todo o processo de desenvolvimento, dentro de um contexto de ciclo de vida iterativo/incremental, a fim de apoiar a construção de sistemas mais seguros.

Apesar da adição inicial de recursos com a utilização do PASS, esse acréscimo pode resultar em produtos mais confiáveis e seguros, que busquem garantir integridade, disponibilidade e confidencialidade das informações e, por conseguinte, clientes mais satisfeitos.

Como conclusões mais importantes deste trabalho, podem-se destacar:

- Implantar práticas de segurança no processo de desenvolvimento de software com o fito de produzir produtos de software mais seguros se mostrou viável e importante.
- Realizar modelagem de ameaças é uma das atividades mais importantes do processo para identificar as vulnerabilidades de segurança reais e auxiliar na identificação dos problemas que causariam maior impacto e promoveriam maior risco.
- Especificar necessidades de segurança conduz à definição de requisitos de segurança, determina a identificação e seleção dos problemas de segurança mais críticos, e encerra-se por meio da aprovação dos requisitos pelo cliente e pelos demais envolvidos.
- Não existem padrões de segurança largamente aceitos a serem utilizados em conjunto com processos de desenvolvimento de software.

Destacam-se como principais contribuições deste trabalho:

- Uma proposta de um processo de desenvolvimento de software formado por atividades de segurança, permitindo uma visão mais detalhada da segurança

de software, pois isto influencia fortemente na produção de software mais seguro.

- Ressaltar a importância de se aplicar boas práticas de segurança da informação no ciclo de vida de desenvolvimento de software.
- Proposição de artefatos de apoio à execução do Processo de Apoio à Segurança de Software.

Não obstante, destacamos que apesar da eficaz aplicação de técnicas e métodos de segurança de software não se pode garantir a prevenção de um software sem erros.

Referências

Alberts, C. *et al*. (2001), "*OCTAVE - The Operationally Critical Threat, Asset, and Vulnerability Evaluation*", Carnegie Mellon – Software Engineering Institute, Disponível em: <www.cert.org/octave>. Acesso em junho de 2006.

Alberts, C. *et al*. (2003), "*OCTAVE-S Implementation Guide*". Carnegie Mellon – Software Engineering Institute. Version 0.9. August 2003. Disponível em: <www.cert.org/octave>. Acesso em Março de 2007.

Andrade, Jeann Marcell Silva. 2005, Avaliação de Processos de Software em Ambientes de Desenvolvimento de Software Orientados à Organização. Dissertação Mestrado em Ciências em Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro (UFRJ). Disponível em: <http://www.cos.ufrj.br/~taba>. Acesso em Maio de 2006.

CMMI, 2006, Capability Maturity Model Integration, Version 1.2. (CMMI-SE/SW, V1.2 – Continuous Representation), SEI Technical Report CMU/SEI-2006-TR-008.

Common Criteria (2005), Version 2.3, August 2005. Disponível em: <http://www.commoncriteriaportal.org>. Acesso em janeiro de 2007.

Howard, M.; LeBlanc D. (2002), Writing Secure Code, 2nd edition. Microsoft Press.

ISO/IEC 12207:1995, Amd 1:2002, (2002), Software engineering Information technology -- Software life cycle processes, ISO – International Organization for Standardization, Geneva, Switzerland.

ISO/IEC 15408-1. (2005a) *Information technology – Security techniques – Evaluation criteria for IT security – Part 1: Introduction and general model.*

ISO/IEC 15408-2. (2005b) *Information technology – Security techniques – Evaluation criteria for IT security – Part 2: Security functional requirements.*

ISO/IEC 15408-3. (2005c) *Information technology – Security techniques – Evaluation criteria for IT security – Part 3: Security assurance requirements.*

ISO/IEC 21827. (2002) *Information technology - Systems Security Engineering - Capability Maturity Model*.

ISO/IEC 27002. (2005) *Information technology – Security techniques – Code of practice for information security management.*

ISSEA. (2003), *International Systems Security Engineering Association*. www.issea.org. Acesso em agosto de 2003.

McGraw, G.; Viega, J., (1999), "Make your *software* behave". Disponível em: <http://www.ibm.com/developerworks/library/s-behave.html>. Acesso em Março de 2006.

McGraw, G. (2004), *Software Security*, IEEE Security and Privacy, March/April 2004, páginas 32-35.

McGraw, G. (2006), Software security: building security in. Addison-Wesley. 1ª Edição.

Paulk *et al*. (1993), *Capability Maturity Model for Software*. Version 1.1 (CMU/SEI-93-TR-024, ADA 263403). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 1993.

PMBOK Guide 2004 Edition. (2004) "A Guide to the Project Management Body of Knowledge", Disponível em <http://www.pmi.org>. Acesso em dezembro 2006.

RUP. (2003) Rational Unified Process®. Rational Software Corporation. Copyright © 1987 - 2003

SSE-CMM. (2003) *System Security Engineering – Capability Maturity Model*, Version 3, Disponível em <www.sse-cmm.org>. Acesso em janeiro de 2006.