GUIA DE PROJETOS DE *SOFTWARE* COM PRÁTICAS DE MÉTODOS ÁGEIS PARA O SISP

(VERSÃO 1.0)

BRASÍLIA
2015

Presidenta da República

Dilma Vana Rousseff

Ministro do Ministério do Planejamento, Orçamento e Gestão

Nelson Barbosa

Secretário de Logística e Tecnologia da Informação

Cristiano Rocha Heckert

Secretário-Adjunto de Logística e Tecnologia da Informação

Fernando Antônio Braga de Siqueira Júnior

Departamento de Governança, Gestão de Pessoas e Sistemas da Informação

Wagner Silva de Araújo

Coordenação-Geral de Sistemas da Informação

Orlando Batista da Silva Neto

Equipe Técnica de Elaboração

Lellis Marçal Mesquita

Jansen Araújo da Fonseca

Equipe de Apoio

Laureano Struck

Lucinéia Turnes

Valéria Maria Siqueira Bezerra

Contribuidores

Para a concepção e construção deste Guia, foram fundamentais a participação e a contribuição com experiências e conhecimentos dos seguintes envolvidos:

BACEN - Banco Central do Brasil

Carla Micheli de Ávila
Eduardo Weller
Javé Barbosa de Meneses
Paulo Ricardo Brazeiro de Carvalho
Tatiana Miranda Gama
Valdir Pereira Macedo

CGU – Controladoria-Geral da União

Tiago Chaves Oliveira

CSJT - Conselho Superior da Justiça do Trabalho

Herbert Bezerra Parente

INEP - Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira

Ladjane Souza de Arruda

IPHAN - Instituto do Patrimônio Histórico e Artístico Nacional

Humberto Mattos Carvalho

MP - Ministério do Planejamento, Orçamento e Gestão

Daniel Portilho Troncoso
Ricardo Miotto Lovatel
Silvia Viviane de Sousa Belarmino
Vinicius de Faria Silva
Waldeck Pinto de Araújo Junior

SERPRO - Serviço Federal de Processamento de Dados

Representantes das Áreas de Negócio, Desenvolvimento e Infraestrutura de TI

TCU - Tribunal de Contas da União

George Atsushi Murakami
Fabiana Ruas

UFMT - Universidade Federal de Mato Grosso

Reni Elisa da Silva Pontes

UNB - Universidade de Brasília

Hilmer Rodrigues Neri
Paulo Roberto Miranda Meirelles

Outros Contribuidores:

Francis Michaell Idzi
Guilherme Siqueira Simões
Renato Willi Silva Consigliero

Normalização Bibliográfica: CODIN/CGPLA/DIPLA

B823g

Brasil. Ministério do Planejamento, Orçamento e Gestão.

Guia de Projetos de Software com práticas de métodos ágeis para o SISP: versão 1.0 / Ministério do Planejamento, Orçamento e Gestão, Secretaria de Logística e Tecnologia da Informação. -- Brasília: MP, 2015.

90 p.: il.

1. Software 2. Tecnologia da Informação 3. Administração Pública

I. Título

CDU 004.4

Este documento encontra-se disponível em http://www.sisp.gov.br

GUIA DE PROJETOS DE *SOFTWARE* COM PRÁTICAS DE MÉTODOS ÁGEIS PARA O SISP

(VERSÃO 1.0)

ACRÔNIMOS

APF - Administração Pública Federal

BACEN - Banco Central do Brasil

CGSIS - Coordenação-Geral de Sistemas de Informação da SLTI

CSJT - Conselho Superior de Justiça do Trabalho

CTI - Comitê de Tecnologia da Informação

CVM - Comissão de Valores Mobiliários

DeGSI - Departamento de Governança e Sistema de Informação

DOD - Documento de Oficialização de Demanda

DTI/SE/MP - Departamento de Tecnologia da Informação vinculada a Secretaria Executiva do Ministério do Planejamento, Orçamento e Gestão

EGP - Escritório de Gerenciamento de Projetos

IN04 - Instrução Normativa SLTI/MP n 04/2014

INEP - Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira

IPHAN - Instituto de Patrimônio Histórico e Artístico Nacional

ISO - *International Organization for Standardization* (Organização Internacional para padronização)

MP - Ministério do Planejamento, Orçamento e Gestão

MDS-SISP - metodologia de desenvolvimento de *software* de referência para os órgãos do SISP

MGP-SISP - Metodologia de Gerenciamento de Projetos do SISP

MGPP-SISP - Metodologia de Gerenciamento de Portfólio de Projetos do SISP

O.S. - Ordem de Serviço

PDTI - Plano Diretor de Tecnologia da Informação

PEI - Planejamento Estratégico Institucional

PMBOK - *Project Management Body of Knowledge* (Corpo de Conhecimento de Gerenciamento de Projetos)

PPA - Plano Pluri Anual

PSW-SISP - Processo de *Software* para o SISP

SISP- Sistema de Administração dos Recursos de Tecnologia da Informação

SEGEP/MP - Secretaria de Gestão Pública do Ministério do Planejamento, Orçamento e Gestão

SERPRO - Serviço Federal de Processamento de Dados

SLTI - Secretaria de Logística e Tecnologia da Informação

STN - Secretaria do Tesouro Nacional

TCU - Tribunal de Contas da União

TI - Tecnologia da Informação

TIC - Tecnologia da Informação e Comunicação

UNB - Universidade de Brasília

XP - *eXtreme Programming* (Programação Extrema)

SUMÁRIO

LISTA DE FIGURAS

LISTA DE QUADROS

PARTE I

1. APRESENTAÇÃO

A Secretaria de Logística e Tecnologia da Informação do Ministério do Planejamento, Orçamento e Gestão tem como missão normatizar, desenvolver e fomentar políticas públicas nas áreas de Logística e Tecnologia da Informação. É também o órgão central do Sistema de Administração de Recursos de Tecnologia da Informação, onde existem mais de duzentos órgãos ou entidades, com diferentes níveis de maturidade em seus processos de tecnologia da informação e comunicação. Na SLTI, cabe ao Departamento de Governança e Sistemas de Informação por meio da Coordenação-Geral de Sistemas de Informação (CGSIS) construir e disponibilizar meios para facilitar a melhoria dos processos de desenvolvimento de *software* nessas instituições.

Atualmente, verifica-se o crescimento da utilização de metodologias ágeis de desenvolvimento de *software* no mercado internacional e nacional motivado pelos valores e princípios do Manifesto Ágil (Agile Manifest, 2001) e pelos vários benefícios encontrados na sua aplicação, como, por exemplo, a maior tempestividade na entrega de resultados que agregam valor ao negócio e aliado a pouca efetividade das contratações de serviços geradas pelo uso do modelo tradicional de desenvolvimento. Instituições públicas como a Secretaria do Tesouro Nacional, Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira, Tribunal Superior do Trabalho, Instituto do Patrimônio, Histórico e Artístico Nacional e Banco Central do Brasil já utilizam metodologias ágeis, tendo alcançado resultados efetivos. Sua aplicação também vem sendo estudada por órgãos de controle, como pode ser visto no Acórdão 2314 de 2013 do Tribunal de Contas da União, onde são levantados riscos de sua utilização na Administração Pública Federal (Acórdão TCU 2314, 2013).

2. ESTRUTURA DO GUIA

Este guia está dividido em três partes e quatro anexos distribuídos da seguinte forma:

Parte I: Contextualização do Guia.

Parte II: Descreve um modelo de referência para construção de projetos de *software* com práticas de métodos ágeis e terceirização do desenvolvimento.

Parte III: Descreve as atividades relacionadas ao modelo de referência para construção de projetos de *software*.

Anexo I: Apresenta conceitos e fundamentos do modelo de referência desde Guia, tais como: conceitos gerais, papéis envolvidos, artefatos e reuniões.

Anexo II: Apresenta um exemplo de itens da cadeia de valor do produto para o modelo de referência de construção de projetos de *software* proposto.

Anexo III: Apresenta uma avaliação da função deste guia em relação aos riscos levantados no acórdão do TCU 2314/2013.

Anexo IV: Apresenta um quadro exemplificativo de não conformidades.

Anexo V: Apresenta um quadro exemplificativo de itens dos tipos de alterações em funcionalidades em métodos ágeis.

3. ESTRATÉGIA ÁGIL DO SISP

A partir do interesse na adoção de métodos ágeis por órgãos ou entidades do SISP, a SLTI por meio da CGSIS criou a Estratégia Ágil para o SISP, que representa um conjunto de ações para viabilizar e difundir o uso dessa metodologia nas instituições públicas. Entre as ações da estratégia ágil está a construção deste Guia. A relação das ações da estratégia ágil é:

- Realizar seminários sobre utilização de métodos ágeis na Administração Pública Federal;

- Disponibilizar informações sobre o uso de métodos ágeis na comunidade "Processo de *Software* para o SISP" do portal do SISP;
- Capacitar servidores de órgãos ou entidades do SISP na utilização de métodos ágeis;
- Promover eventos de disseminação da cultura ágil, seus valores, princípios, premissas e benefícios nas instituições públicas;
- Realizar e disponibilizar benchmarking (comparações) do uso de métodos ágeis no governo;
- Construir e disponibilizar um guia de projetos de *software* com práticas de métodos ágeis para órgãos ou entidades do SISP.

No portal do SISP (www.sisp.gov.br) na comunidade de "Processo de *Software* para o SISP", foram disponibilizados os relatórios dos benchmarking realizados, e o conteúdo do 1° Seminário ágil realizado.

Para viabilizar a construção deste Guia, foi criado um grupo de trabalho, no qual foram apresentadas, discutidas e aprovadas propostas de estruturação deste Guia com interessados no assunto.

4. ESCOPO DO GUIA

O foco deste guia é somente a construção de projetos de *software*, não estão incluídos processos nem atividades para a realização de sustentação de sistemas de informação já existentes da instituição, tais como a realização de pequenas correções ou evoluções de curto prazo que não demandam um projeto.

A demanda por um projeto de *software*, em geral, nasce na área de negócio, passa pela gestão estratégica e de portfólio de projetos, onde sua viabilidade é avaliada, posteriormente são realizados os critérios de priorização e balanceamento, permitindo que projetos de maior valor para a organização sejam aprovados. Uma vez aprovado, o projeto entra na etapa de execução, ou seja, inicia-se a sua construção.

Este Guia apresenta um modelo de referência para esta última etapa do projeto. Entende-se como modelo de referência um documento orientador e não prescritivo, de forma a manter a autonomia dos projetos e times, que podem ser adaptados conforme suas características e particularidades, fomentando a melhoria contínua da aplicação de métodos ágeis. Este modelo de referência é ideal para instituições que possuem um quadro de pessoal reduzido e para equipes de servidores responsáveis pela gestão da T.I. e fiscalização de contratos.

As atividades de construção de projetos, adotadas neste guia, permitem a realização de entregas incrementais de *software* através de *releases*. Também são disponibilizadas atividades de acompanhamento, controle e fiscalização de projetos, extraídas da legislação atual e de contratações de TI.

5. PROJETO DE *SOFTWARE*

Projeto de *software* é um serviço disponibilizado pela Área de Tecnologia da Informação para atender várias necessidades do órgão ou entidade. São exemplos de um projeto de *software*:

- **Novo Desenvolvimento:** Um novo *software* ou sistema de informação pode ser desenvolvido integralmente ou reconstruído a partir de um legado;
- **Customização e Implantação:** A partir de novas necessidades de negócio, podem ser encontrados sistemas de informação já existentes para estas necessidades, dispensando a construção de um novo. Nestes casos o projeto pode ser a customização e implantação do sistema dentro da estrutura e arquitetura da instituição, utilizando um código-fonte que foi, em todo ou em parte, cedido ou repassado à instituição, ou obtido por outros meios;
- **Manutenções Evolutivas:** Em geral as demandas de evolução de sistemas de informação são de responsabilidade da

equipe de sustentação. Estas demandas quando de tamanho ou complexidade maior poderão ser tratadas como projetos de *software*. Manutenções adaptativas, perfectivas e cosméticas são exemplos de evoluções de sistemas de informação.

Os projetos de *software* não estão restritos a lista apresentada, cada área de tecnologia da informação do órgão ou entidade do SISP pode estabelecer critérios específicos de criação de projetos, de acordo com as necessidades e demandas da instituição e capacidade de atendimento da TI.

6. TERCEIRIZAÇÃO DE SERVIÇOS DE TI

A terceirização de serviços na Administração Pública Federal é amparada pelo Decreto-lei N° 200, de 25 de fevereiro de 1967 (desobrigação de tarefas executivas). Posteriormente, por meio do Decreto nº 2.271, de 7 de Julho de 1997, regulamentou-se a diretriz de execução indireta contida no § 7° do Art. 10 do Decreto-lei 200/1967, que incluiu os serviços de informática no rol de serviços com execução indireta. Mais recentemente, a Instrução Normativa 04 do Ministério do Planejamento, Orçamento e Gestão regulamentou a execução indireta de serviços de TI à luz da legislação corrente, e das leis 8.666/1993 e 10.520/2002.

Nos órgãos ou entidades do SISP, podem ser encontradas áreas de Tecnologia da Informação com diferentes níveis de maturidade e capacidade de execução de serviços necessários a produção de *software*. Alguns dos principais serviços envolvidos na construção de projetos de *software* e favoráveis à terceirização são:

- **Serviços de Desenvolvimento de *Software*:** envolvem atividades típicas de engenharia de *software* como elicitação de requisitos, implementação, testes e implantação de sistemas de informação. Este serviço é conhecido como fábrica de *software*;
- **Serviços de Apoio à Avaliação de Qualidade:** envolvem tarefas de avaliação de qualidade de *software* e seus artefatos, tais como, realizar testes, avaliar arquitetura, modelos de dados, documentação, entre outros. Este serviço é conhecido como fábrica de qualidade ou fábrica de testes;
- **Serviços de Apoio às Métricas de *Software*:** envolvem tarefas de estimativa, contagem e validação do tamanho funcional de sistemas de informação. Este serviço é conhecido como fábrica de métricas e, em geral, utiliza a métrica de Pontos de Função;
- **Serviços de Apoio ao Gerenciamento de Projetos de *Software*:** envolvem tarefas de acompanhamento e controle de projetos. Esse serviço descreve atividades típicas de um Escritório de Gerenciamento de Projetos (EGP);
- **Serviços de Infraestrutura de TI:** envolvem atividades de configuração, disponibilização e administração de infraestrutura física e lógica de Centro de Dados, necessária à produção continuada de sistemas de informação, bem como ao armazenamento de dados, incluindo a guarda e a recuperação dos dados dos sistemas.

É importante lembrar que para licitar um serviço, além das necessidades internas da instituição, devem ser avaliados também critérios econômicos envolvidos na contratação (Lei 8.666, 1993).

Visto que a maioria dos órgãos ou entidades do SISP possui pequenas equipes internas, normalmente insuficientes para executar projetos de desenvolvimento e sustentação de sistemas de informação. Este guia apresenta um modelo de construção de projetos de *software* baseado na terceirização desses serviços.

Por uma questão de escopo, optou-se por um guia que trata inicialmente de contratação somente dos "*Serviços de Desenvolvimento de Software*". Em outro momento, a SLTI poderá ampliar o escopo para boas práticas de contratação de outros serviços, tais como o de apoio à avaliação de qualidade e métricas de *software*.

7. ENVOLVIDOS NA CONSTRUÇÃO DE PROJETOS DE *SOFTWARE*

Várias áreas dos órgãos ou entidades do SISP estão envolvidas na construção de projetos de *software*, podendo ser envolvida uma empresa contratada para realizar serviços técnicos de produção do *software*. Em geral as áreas envolvidas são: área requisitante da solução, área de tecnologia da informação e área administrativa.

A contratada de desenvolvimento de *software*, única terceirizada envolvida neste guia, possui suas atividades distribuídas em práticas dos métodos ágeis para o desenvolvimento de *software*.

Segue detalhamento de envolvidos em projetos de *software*:

- **Área Requisitante do Projeto de *Software*:** unidade do órgão ou entidade que demande a construção de um projeto de *software*. É responsável pela definição dos requisitos, priorização de funcionalidades, homologação negocial e apoio constante aos projetos de *software*;
- **Área de Tecnologia da Informação:** unidade setorial ou seccional do SISP, bem como área correlata, responsável por gerir a Tecnologia da Informação do órgão ou entidade (IN04 SLTI/MP, 2014). Dentro da área de tecnologia da informação os principais envolvidos na construção de projetos de *software* são:
  - **Unidade Setorial de Sistemas:** setor de TI responsável pelo desenvolvimento, medição e avaliação de qualidade de *software*, principalmente no aspecto técnico;
  - **Unidade Setorial de Infraestrutura de TI:** setor responsável pela infraestrutura de TI. Faz a criação e suporte dos ambientes de TI utilizados em projetos e sustentação de sistemas de informação. Alguns órgãos ou entidades do SISP terceirizam os serviços de infraestrutura de TI.
  - **Escritório de Projetos de TI:** O Escritório de Gerenciamento de Projetos (EGP) ou Escritório de Projetos é uma estrutura, função ou unidade organizacional que centraliza e coordena o gerenciamento de projetos sob seu domínio (MGP-SISP, 2011). As atividades sugeridas neste guia são a abertura, fechamento, acompanhamento e distribuição de informações dos projetos de *software*;

- **Contratada de Desenvolvimento de *Software*:** empresa contratada pelo órgão ou entidade do SISP, responsável pelas atividades técnicas de desenvolvimento de *software*, tais como análise, projeto, implementação, testes e implantação;
- **Área Administrativa:** unidade setorial e seccional do Sistema de Serviços Gerais – SISG – com competência para planejar, coordenar, supervisionar e executar as atividades relacionadas aos processos de contratação e procedimentos administrativos de contratos (IN04 SLTI/MP, 2014). Apoia na gestão e fiscalização do contrato de desenvolvimento de *software* com práticas de métodos ágeis, sugerido neste guia.

Cada órgão ou entidade do SISP pode apresentar diferentes variações de envolvidos na construção de projetos de *software*. Utilizou-se essa estrutura porque ela reúne condições que favorecem a entrega de projetos a clientes e usuários, dentro de prazos, custos e qualidade esperados.

8. RELAÇÃO DESTE GUIA COM O "PROCESSO DE *SOFTWARE* PARA O SISP"

No portal do SISP (www.sisp.gov.br), está disponível o "Processo de *Software* para o SISP" (PSW-SISP, 2012). O "Processo de *Software* para o SISP (PSW-SISP)" é bastante amplo e baseado em metodologias tradicionais de desenvolvimento de *software*. Ele foi disponibilizado para orientar e elevar os níveis de maturidade dos órgãos ou entidades do SISP em processos de gestão estratégica, gestão de projetos, gestão de segurança, engenharia de *software*, produção colaborativa, gestão de contratação, gestão de infraestrutura e sustentação de sistemas de informação. São objetivos do PSW-SISP:

- Evoluir de forma conjunta as soluções;
- Utilizar de forma responsável e mais eficiente os recursos públicos;

- Elevar os níveis de qualidade e controle das soluções;
- Alinhar as soluções ao planejamento estratégico;
- Reter a inteligência das soluções nos órgãos;
- Padronizar processos e artefatos.

Assim sendo, o ideal é que o processo seja usado conforme as necessidades e maturidade do órgão ou entidade, em se tratando de desenvolvimento de *software* e gestão de projetos. Os órgãos ou entidades devem decidir quais as atividades são adequadas à maturidade e ao projeto em desenvolvimento ou manutenção (evolutiva e corretiva), sendo que ele considera algumas atividades mínimas como essenciais para a qualidade do *software*.

Este Guia foca a construção de projetos de *software*, utilizando práticas de métodos ágeis, e pode ser utilizado de forma complementar ao "Processo de *Software* para o SISP". Algumas atividades deste Guia podem ser utilizadas dentro de determinados processos do PSW-SISP, tais como:

- 4.1 Executar projeto;
- 4.2 Monitorar e controlar o trabalho do projeto;
- 4.3 Preparar ambiente de homologação;
- 4.4 MDS-SISP;
- 4.5 Gerenciar contratação.

9. PRÁTICAS DE MÉTODOS ÁGEIS

Muitas instituições, em geral, iniciam o uso de métodos ágeis por meio da adoção de *Scrum*, porque descreve uma boa estratégia para a gestão de projetos de *software*. No entanto, *Scrum* é apenas uma parte do que é necessário para oferecer soluções sofisticadas para seus *stakeholders*. Inevitavelmente, as equipes precisam olhar para outros métodos para preencher lacunas que o *Scrum* não considera propositadamente (*Disciplined Agile Delivery*, 2014).

Para Fadel e Silveira (2010), a metodologia *Scrum* não define uma técnica específica para o desenvolvimento de *software* durante a etapa de implementação, ele se concentra em descrever como os membros da equipe devem trabalhar para produzir um sistema flexível, num ambiente de mudanças constantes.

Segundo Franco (Franco, 2007), a metodologia *Extreme Programming* (XP) surgiu como uma tentativa para solucionar os problemas causados pelos ciclos de desenvolvimento longos dos modelos de desenvolvimento tradicionais. O *XP* é composto por práticas que se mostraram eficientes nos processos de desenvolvimento de *software* (Beck, 1999).

O *Lean* não é uma prática de desenvolvimento ou um gerenciamento de projetos, mas sim um conjunto de princípios, valores e ferramentas que tornam o desenvolvimento enxuto (Fadel e Silveira, 2010).

Neste guia será apresentado um modelo de referência de construção de projetos de *software* que utiliza práticas do *Scrum*, *XP* e *Lean* organizadas de forma complementares.

10. UTILIZAÇÃO DESTE GUIA

Este guia é direcionado para órgãos ou entidades do SISP, principalmente as instituições que dispõem de pequenas equipes de servidores para gestão de projetos de *software* nas áreas de TI.

10.1. POR QUE UTILIZAR ESTE GUIA

Este guia foi estruturado com o intuito de apoiar órgãos ou entidades do SISP, em tarefas como:

- Construir ou customizar seus modelos de construção de projetos de *software*, baseados em práticas de métodos ágeis;
- Servir como modelo de execução de serviços de desenvolvimentos de *software*;
- Auxiliar na gestão de projetos de *software* desenvolvidos com práticas de métodos ágeis;
- Orientar a definição ou adequação de atividades e práticas que auxiliam no gerenciamento e fiscalização de projetos e contratos de desenvolvimento de *software*.

É importante ressaltar que este guia disponibiliza uma série de boas práticas para o desenvolvimento de projetos de *software* com práticas de métodos ágeis, aplicadas na contratação, e cada instituição do SISP deve customizar as atividades e recomendações do guia, conforme a sua realidade.

10.2. O QUÊ O GUIA NÃO RESOLVE ?

Este subtópico relaciona algumas limitações impostas pela concepção deste Guia. Os temas abaixo relacionados ou não foram tratados, ou resolvidos ou exauridos:

a) Gestão de riscos de projetos de *software*;
b) Seleção de ferramentas, técnicas ou métodos para Engenharia de *Software*;
c) Seleção de fornecedores para contratação de empresa de desenvolvimento de *software*;
d) Transição entre metodologias ou paradigmas de desenvolvimento de *software*;
e) Equacionamento de problemas ou vícios de uma contratação existente para desenvolvimento de *software*;
f) Métricas de *software*. O Guia não trata de mensuração de *software*: tamanho funcional X valor agregado X esforço;
g) Não esgota determinados aspectos da Cultura Ágil;
h) Implantação de governança corporativa e de tecnologia da informação;
i) Não resolve todos problemas de gestão de projetos de TI, mas é extensível e adaptável;
j) Não mitiga riscos da contratação de empresa de desenvolvimento de *software*;
k) Não contempla todos os aspectos do gerenciamento de um projeto, tais como aquisições, seleção e aprovação de projetos de *software*.

Entretanto, este Guia sugere algumas ideias e subsídios para gestão de riscos, elaboração de termo de referência, disseminação da cultura ágil no SISP, entre outros.

11. GRUPOS DE ATIVIDADES

O modelo de execução de projetos de *software* apresentado neste guia, baseia-se no agrupamento de atividades similares de produção de *software*. Nortearam a adoção deste modelo, principalmente, metodologias de desenvolvimento ágeis de instituições públicas, legislações de contratações de TIC, definições extraídas do grupo de trabalho, publicações e *frameworks* de desenvolvimento ágil de *software*, experiências práticas de estudiosos, profissionais e empresas que utilizam métodos ágeis. A seguir são apresentados os grupos de atividades:

- **Grupos de Atividades de Construção de Projeto:** representa agrupamentos de atividades que são realizadas para planejar o projeto, construir e entregar *releases* do produto de *software*,são eles:
  - **Atividades de Planejamento:** contempla concepção do Documento de Visão, em que são descritos objetivos de negócio e características-chaves (*features*) do produto e sua distribuição em diferentes *releases*, formando um *roadmap* do produto;

  - **Atividades de Construção do *Release*:** contempla o planejamento, especificação e implementação dos objetivos de negócio e características-chaves do produto contidas nos *releases*, distribuídas em iterações (*Sprints*). São atividades definidas a partir de práticas de métodos ágeis, como *Scrum*, *XP* e *Lean*;
  - **Atividades de Transição:** contempla atividades que garantam a implantação de cada *release*, avaliação dos resultados obtidos e das condições de entrega, e suporte do produto do projeto;
- **Atividades de Gestão de Ordem de Serviço:** apresenta atividades de gestão de ordens de serviço para o contrato de desenvolvimento de *software*, aplicada aos grupos de atividades de construção do projeto;
- **Atividades de Acompanhamento do Projeto:** contempla atividades de acompanhamento da execução do projeto, de forma que possíveis problemas possam ser identificados no momento adequado e que possam ser tomadas ações preventivas e corretivas, quando necessário, para controlar a execução do projeto. Estas atividades contemplam atualização dos status do portfólio de projetos, identificação e comunicação de desvios e impedimentos e apoio à equipe do projeto para garantir entregas definidas;
- **Atividades de Gestão de Ambientes de TI:** contempla atividades para viabilizar a criação de ambientes de TI e facilitar a sua disponibilidade para construir e implantar cada *release* do produto de *software*.

A figura 1 descreve o relacionamento entre os grupos de atividades, distribuídos entre as áreas envolvidas em projetos de *software*. A estratégica da instituição direciona a aprovação do projeto pelo Comitê de Tecnologia da Informação por meio de critérios de seleção e priorização.

A partir dos instrumentos estratégicos e de planejamento, mencionados na figura 1, a área de negócio com apoio da área tecnológica elabora o documento de visão que contempla aspectos relacionados a problemas, objetivos de negócio, necessidades, processos de negócio, expectativas, entre outros e, por fim, registra uma proposta de solução, a qual envolve elementos tecnológicos, tais como interoperabilidade com outros sistemas de informação baseados em computador (relação de consumo ou fornecimento), descrição das características-chaves do produto, descrição dos consumidores da solução (humanos ou não humanos), descrição dos requisitos de ambientes, descrição dos requisitos da documentação, descrição dos requisitos do produto, menção de tecnologias importantes, entre outros.

A área tecnológica apoia a elaboração do documento de visão. O documento de visão pode reunir e unificar a visão do produto e a visão da solução em um mesmo documento. A contratada de desenvolvimento de *software* poderá auxiliar no levantamento e especificação dos requisitos de *software*.

O grupo de atividades da gestão de ordem de serviço, ilustrado na figura 1, reúne práticas importantes para controle, fiscalização e monitoração da execução e dos produtos entregues pela contratada de desenvolvimento de *software*. Por meio dela é possível mitigar riscos da contratação, tais como falta de qualidade técnica e funcional, pagamento sem produto de *software* funcional, pagamento sem métrica objetiva, entre outros.

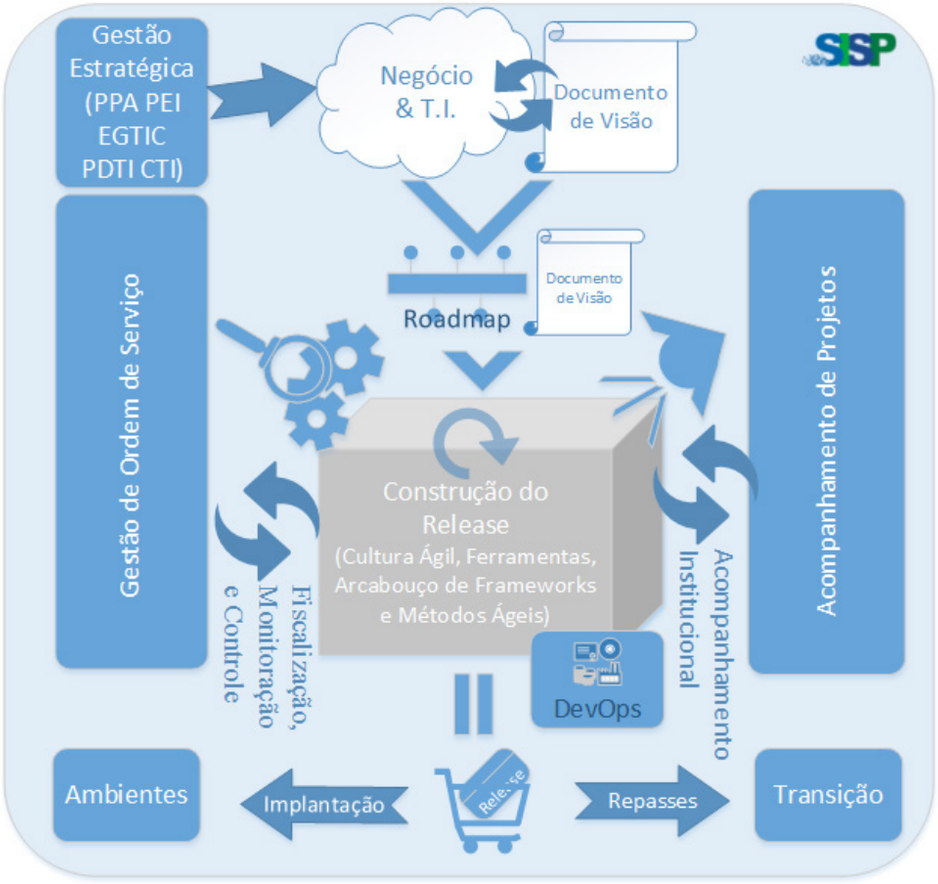

Figura 1. Fluxo de Relacionamento entre Grupos de Atividades do Projeto.

O grupo de atividades de acompanhamento do projeto, mostrado na figura 1, reúne atividades para monitorar institucionalmente os projetos em execução pela contratada. Riscos relacionados ao projeto são mitigados, tais como desvios da execução, impedimentos, descumprimento do cronograma, produto não funcional, entre outros.

O grupo de atividades de construção do *release*, representado pela caixa cinza da figura 1, é realizado predominantemente pela contratada e contempla algumas atividades em que a instituição participa no momento da execução. Alguns controles são exercidos em algumas atividades tais como validação do incremento de *software* e planejamento. No entanto, a instituição não deve se envolve na microgestão da execução de determinadas atividades, tais como, reunião e planejamento diários. A construção do *release* é controlada pela instituição, apenas em determinadas atividades.

O Guia não esgota nem define um arcabouço de *frameworks*, métodos ágeis e cultura ágil.

Por fim, para cada grupo de atividades de construção do projeto (planejamento, construção do *release* e transição) são disparadas atividades de gestão de ordem de serviço, acompanhamento do projeto e gestão dos ambientes de TI. Alguns artefatos produzidos nos grupos de atividades também estão ilustrados na figura 1.

PARTE II

12. MODELO DE REFERÊNCIA PARA CONSTRUÇÃO DE PROJETOS DE *SOFTWARE*

A Figura 2 disponibiliza o modelo de referência para construção de projeto de *software* com práticas de métodos ágeis e terceirização do desenvolvimento, proposto neste guia. Este modelo é formado pela reunião dos grupos de atividades descritos anteriormente.

Os instrumentos de planejamento estratégico da instituição, ilustrados no início da Figura 2, proveem meios de seleção e aprovação dos projetos de maior valor para a organização, são eles: Planejamento Estratégico Institucional - PEI, Estratégica Geral de Tecnologia da Informação e Comunicação - EGTIC, Plano Diretor de Tecnologia da Informação - PDTI e Comitê de Tecnologia da Informação - CTI. Esses instrumentos de planejamento estratégico não foram descritos no modelo.

Na Figura 2 cada grupo de atividades do modelo de referência é apresentado por uma cor específica. Os grupos de atividades específicas de construção do projeto de *software* foram hachurados nas seguintes cores: planejamento na cor azul escuro, construção do *release* na cor verde e transição na cor roxa. As atividades de acompanhamento de projetos estão na cor amarela, gestão de ambientes de TI na cor cinza e gestão de ordem de serviço na cor vermelha.

A organização e estruturação dos grupos de atividades de construção do projeto (planejamento, construção do *release* e transição) e gestão de ambientes de TI, foram definidas, principalmente, com base nas seguintes referências:

- Metodologias ágeis de instituições públicas:
  - PDS-BC Ágil - Processo Ágil de Desenvolvimento de *Software* do Banco Central (PDS-BC Ágil, 2014);
  - MGDS - Metodologia de Gestão e Desenvolvimento de Sistemas do INEP (MGDS-INEP, 2013);
  - MIDAS - Metodologia Iphan de Gestão de Demandas de Desenvolvimento Ágil de *Software*s (MIDAS-IPHAN, 2013);
  - Disponíveis em: http://www.sisp.gov.br/dotlrn/clubs/processodesoftwaresisp/file-storage/index?folder_id=15123068
- *Disciplined Agile Delivery* (*Disciplined Agile Delivery*, 2014);
  - Disponível em: http://www.disciplinedagiledelivery.com/introduction-to-dad/
- *Scaled Agile Framework* (*Scaled Agile Framework*, 2014):
  - Disponível em: http://www.scaledagileframework.com/
- Guia do *Scrum* (Guia do *Scrum*, 2013):
  - Disponível em: http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-Portuguese-BR.pdf
- Agile Alliance Guide (Agille alliance Guide, 2015):
  - Disponível em: http://guide.agilealliance.org/subway.html
- *Scrumex* - Expandindo a Aplicação e os Benefícios do *Scrum* (*Scrumex*, 2014);
  - Disponível em: http://www.scrumex.com.br/blog/

As atividades de gestão de ordem de serviço foram baseadas em referências atuais da legislação de contratações de TI, tais como: Instrução Normativa MP/SLTI Nº 04, de 11 de setembro de 2014, Instrução Normativa N° 02 MP/SLTI, de 30 de abril de 2008 , Guia de Boas Práticas de Contratações de TI (Guia SLTI/MP, 2014), Lei no 8.666, de 21 de junho de 1993, Lei no 10.520, de 17 de junho de 2002 e Processo de Contratação de Serviços de TI para Organizações Públicas (Cruz et al, 2011).

As atividades de acompanhamento do projeto, do modelo de referência, foram extraídas da Metodologia de gerenciamento de Portfólio de Projetos do SISP (MGPP-SISP, 2013) e Metodologia de Gerenciamento de Projetos do SISP (MGP-SISP, 2011) e de técnicas ágeis como gráfico de *burndown*. Também foi incluído *coaching* neste grupo de atividades.

Cada atividade citada no modelo de referência da figura 2 será descrita separadamente na parte III deste guia.

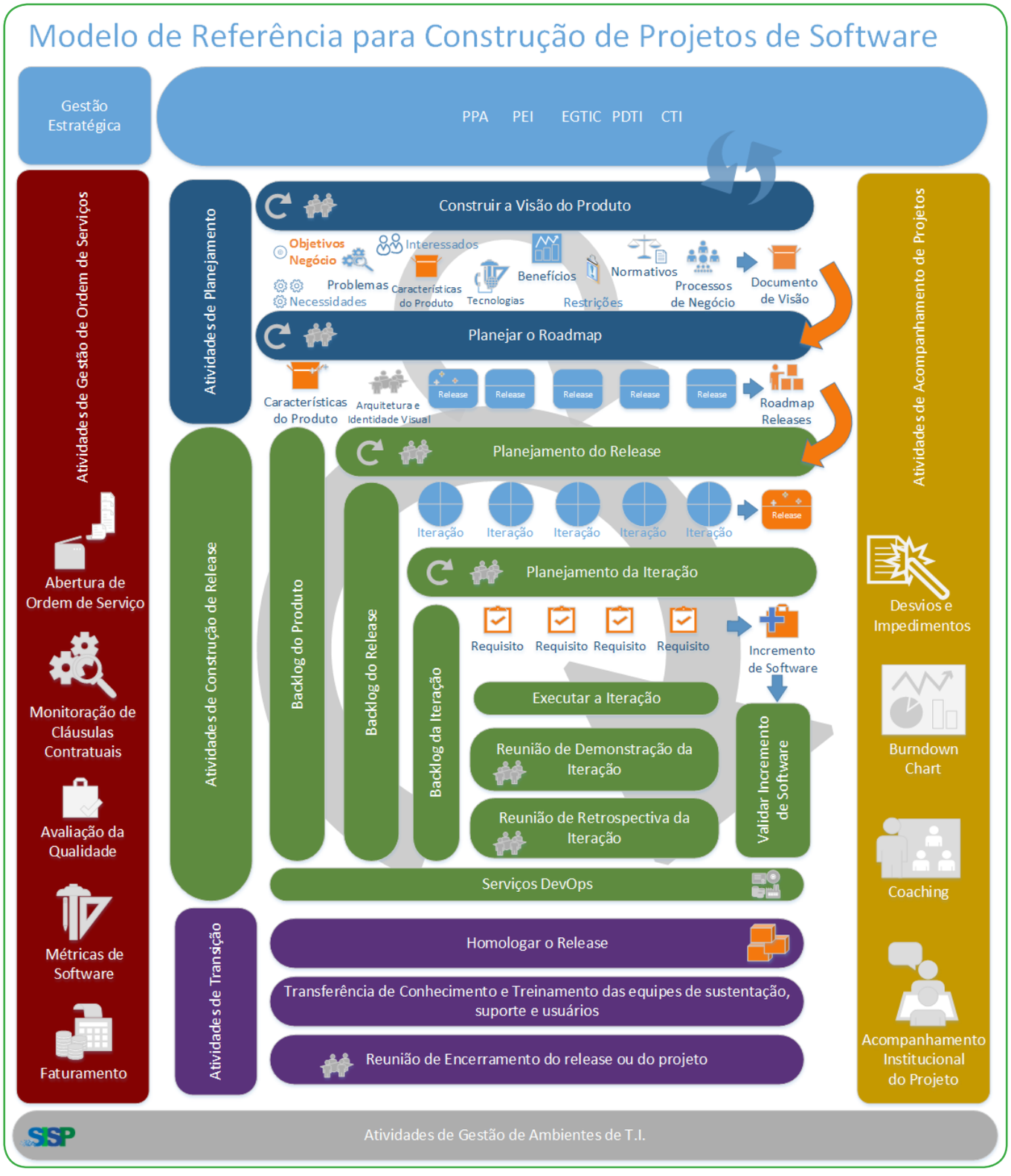

Figura 2. Modelo de Referência para Construção de Projetos de *Software*.

13. PAPÉIS RESPONSÁVEIS POR ATIVIDADES DO MODELO DE REFERÊNCIA

Os quadros 1 e 2 descrevem os papéis ou perfis dos responsáveis por atividades do modelo de referência, apresentado na figura 2. Um papel é responsável por uma ou mais atividades e algumas atividades possuem mais de um responsável. Cada papel pode ser da contratante - órgão ou entidade do SISP, ou da contratada de desenvolvimento de *software*. Os papéis foram extraídos principalmente da metodologia de desenvolvimento ágil *Scrum*, Instrução Normativa 04 do MP, metodologia de gestão de projetos do SISP e das áreas da instituição envolvidas em projetos de *software*. O quadro 2 descreve os perfis da contratada de desenvolvimento de *software* e o quadro 1 da contratante, ou seja, do órgão ou entidade do SISP.

Quadro 1. Papéis da Contratante - Órgão ou Entidade do SISP.

Nome do Papel	Atividades que é Responsável ou que participa como um dos Responsáveis pela Execução
Gestor do Contrato de Desenvolvimento de *Software* (Servidor da Área Requisitante do Projeto ou da Área de TI)	• Abrir Ordem de Serviço; • Tratar Faturamento de Ordem de Serviço; • Monitorar e Controlar Obrigações Advindas de Cláusulas Contratuais; • Fechar Ordem de Serviço.
Fiscal Técnico do Contrato de Desenvolvimento de *Software* (Servidor da Área de TI)	• Avaliar, Aceitar ou Rejeitar Serviços de OSs; • Receber Serviços de OSs.
Fiscal Administrativo do Contrato de Desenvolvimento de *Software* (Servidor da Área Administrativa)	• Tratar Faturamento de Ordem de Serviço; • Monitorar e Controlar Obrigações Advindas de Cláusulas Contratuais.
Dono do Produto / Fiscal Requisitante do Contrato de Desenvolvimento de *Software* (Servidor(es) da Área Requisitante do Projeto)	• Construir a Visão do Produto; • Planejar o *Roadmap*; • Elaborar o *Backlog* do Produto; • Planejar o *Release*; • Validar Incremento de *Software*; • Homologar o *Release*; • Avaliar, Aceitar ou Rejeitar Serviços de OSs; • Preparar e Realizar Treinamentos.
Líder de Projeto (Servidor do Escritório de Projetos)	• Atualizar Acompanhamento do Projeto; • Reunião de Encerramento do *Release* ou Projeto; • Preparar e Realizar Treinamentos.
Coach (Servidor da Área Requisitante do Projeto ou da Área de TI)	• Realizar *Coaching*.
Analista de Métricas (Servidor da Área de TI)	• Medir o *Software*.
Analista de Infraestrutura de TI (Servidor da Área de TI)	• Preparar e Manter Ambientes de TI; • Implantar *Software*.

Quadro 2. Papéis da Contratada para Desenvolvimento de *Software*.	
Nome do Papel	**Atividades que é Responsável ou que participa como um dos Responsáveis pela Execução**
Mestre *Scrum*	• Reunião de Planejamento da Iteração; • Executar a Iteração; • Reunião de Demonstração da Iteração; • Reunião de Retrospectiva da Iteração; • Atualizar Gráfico de *Burndown*.
Equipe de Desenvolvimento	• Reunião de Planejamento da Iteração; • Executar a Iteração; • Reunião de Demonstração da Iteração; • Corrigir Não Conformidades da Ordem de Serviço.

É importante destacar que existem outros papéis, além dos citados nos quadro 1 e 2, que participam das atividades propostas neste guia. Esses papéis são descritos no guia, as vezes, por um perfil agrupador ou descritos separadamente. Por exemplo, analistas de qualidade e equipe de desenvolvimento podem ser compostos por administradores de banco de dados, arquitetos de *software*, analistas de sistemas e outros. Cada instituição, que utilize este guia, pode definir outros papéis além dos descritos, desde que não conflite com a legislação vigente. No anexo I são descritos os papéis e suas responsabilidades gerais.

14. PRINCÍPIOS GERAIS

Os princípios elencados na enumeração abaixo devem ser utilizados, como princípios, nas atividades deste Guia para as seguintes finalidades: priorização, estimativa de esforço, estimativa de prazos e direcionamentos da organização para a equipe de projetos.

14.1. AGREGAÇÃO DE VALOR

O valor agregado é a percepção pelo dono do produto de que um objetivo específico do negócio foi satisfeito. Certifique-se que os acordos firmados e os produtos entregues agregam valor ao negócio.

14.2. SIMPLICIDADE

Elimine tudo aquilo que for desnecessário, pois ele onera e sobrecarrega o processo, as pessoas e o produto.

14.3. FOCO

Estabeleça um foco nos itens de solução, nas necessidades e nos problemas, para que a energia de trabalho se concentre apenas nos objetivos e acordos firmados.

15. FLUXO DE VALOR DO PRODUTO

A finalidade da construção de projetos de *software* é a entrega de um produto (ou versão funcional do produto de *software*) e a finalidade do produto é a entrega de valor agregado para o cliente. A entrega de valor é efetivada, através do atendimento

das necessidades e expectativas do cliente. Os grupos de atividades de planejamento, construção do *release* e transição da figura 2 apresentam um fluxo de valor do produto de *software*, baseado na proposta descrita em *Scrumex* (*Scrumex*, 2014). Itens que determinam o fluxo de valor estão destacados na figura 2 pela cor alaranjada (objetivos de negócio, características-chaves do produto, requisitos, incremento de *software* e *release*).

O modelo de construção de projetos de *software*, proposto neste guia, é realizado através de ciclos de entregas incrementais, por meio de *releases*. Os ciclos de entrega envolvem os grupos de atividades de planejamento, construção do *release* e transição, um para cada *release*. Após a finalização do grupo de atividades de construção do *release* é feita a transição dos produtos para o ambiente de produção.

Nos tópicos abaixo serão descritos artefatos e informações que determinam o fluxo de valor do produto de *software* que permeia cada grupo de atividades de construção do projeto do modelo de referência da figura 2.

15.1. FLUXO DE VALOR ENTRE VISÃO E *ROADMAP* DO PRODUTO

O fluxo de valor do produto de *software* inicia-se no grupo de atividades de planejamento (construir a visão do produto e planejar o roadmap - cor azul escuro da figura 2), o qual consiste na identificação e compreensão das necessidades, problemas, objetivos de negócio, características-chaves do produto (*software features*), oportunidades, desafios e expectativas dos clientes. Também se Inclui neste grupo a elaboração do *roadmap* de *releases*. Ver a apresentação na figura 2.

A compreensão do atual funcionamento dos processos de negócio é importante para capturar as problemáticas e insatisfações do cliente para, finalmente, identificar as macrofunções de uma possível solução. As características-chaves ou macrofunções são atributos de valor ou serviços providos pela solução, derivados dos objetivos de negócio, para satisfazer necessidades de negócio dos clientes, em outros termos, são os aspectos mais relevantes do produto de *software* que o cliente valorizará com mais facilidade.

O artefato visão do produto reúne todo este conhecimento adquirido por meio de técnicas de elicitação de requisitos de *software* e análise de negócio. Para o mapeamento dos *releases* no *roadmap*, também é importante a discussão prévia da arquitetura inicial e a identidade visual do *software* a fim de identificar itens de *backlog* necessários para a primeira entrega. A Figura 3 ilustra o fluxo de valor entre visão do produto e do *roadmap*.

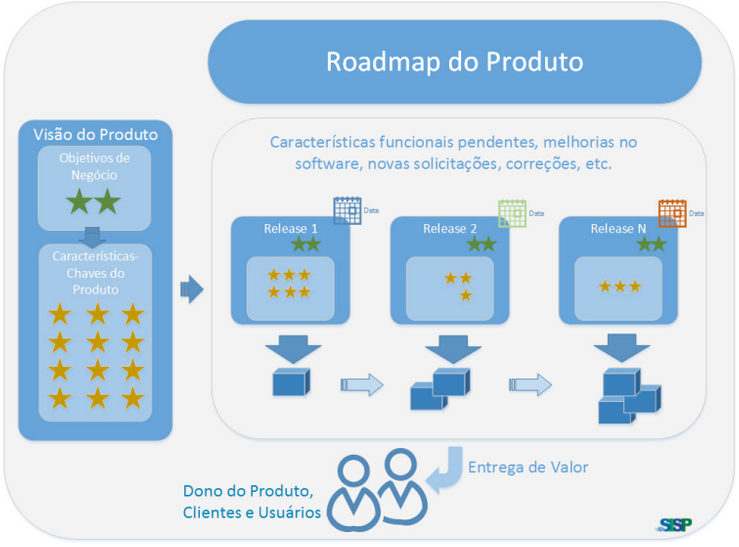

Figura 3. *Roadmap* do Produto.

Após a concepção do Documento de Visão, inicia-se o planejamento de cada entrega (*release*) o qual consiste no agrupamento de objetivos de negócio e características-chaves do produto por meio de critérios de priorização e de importância.

Cada *release* é o lançamento de uma determinada versão de *software* incrementada com novas funcionalidades, as quais satisfazem determinados objetivos de negócio e características-chaves do produto de *software*. O registro cronológico de *releases*, assim como objetivos de negócio e características-chaves associadas, poderão ser consolidadas em um *roadmap* de entregas (planejamento de liberação de versões).

15.2. FLUXO DE VALOR NO PLANEJAMENTO DO RELEASE

O planejamento do *release* está contido no grupo de atividades de construção do *release* (atividades de cor verde da figura 2). O início de cada planejamento do *release* é marcado pela elaboração do *backlog* do produto, técnica extraída do *Scrum*. O *backlog* desdobra objetivos de negócio e características-chave do produto em requisitos de *software* (funcionais e não funcionais) e tarefas técnicas importantes para a construção de produtos de *software*. Para cada *release*, uma ou mais iterações deverão ocorrer. Este fluxo de valor está ilustrado na figura 4.

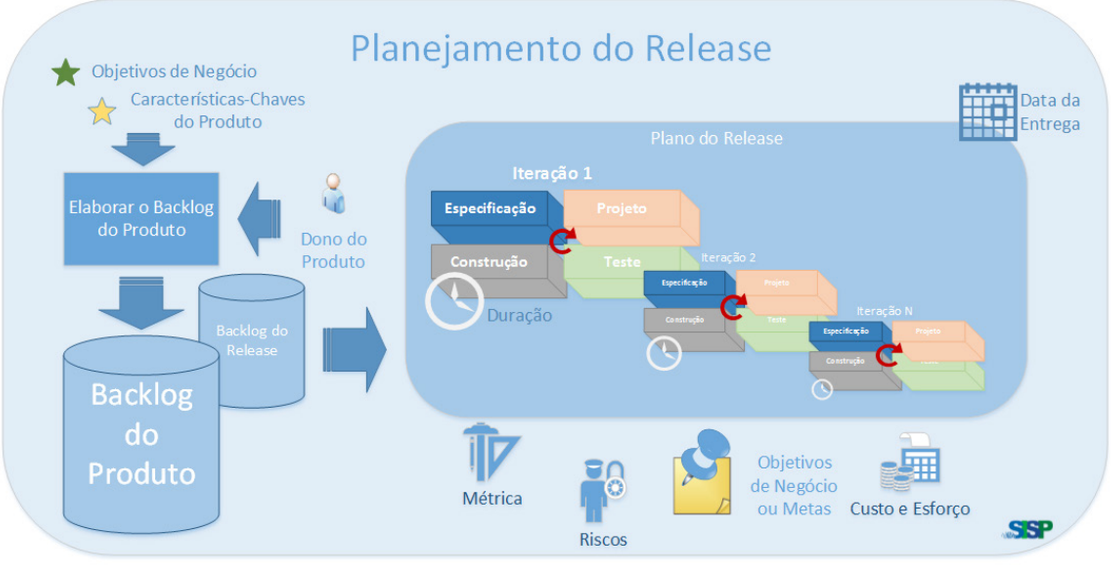

Figura 4. Planejamento do *Release*.

O planejamento do *release*, portanto, consiste em desdobrar objetivos de negócio e características-chaves do produto, correspondidas por itens de *backlog*. Os itens de *backlog* podem pertencer a um ou mais *backlog* de iterações. Eles são orientados por critérios de priorização de seleção de requisitos e cronograma do projeto. Nessa fase também devem ser definidos os conceitos de pronto e preparado, e qual será a duração fixa das iterações. Esse planejamento define o plano do *release*.

15.3. FLUXO DE VALOR NA ITERAÇÃO

Na figura 2, nas atividades de construção do *release*, cada Iteração se inicia com planejamento específico, conforme a figura 5, onde é gerado o plano da iteração, com o objetivo de gerar um incremento de *software*. Os requisitos ou histórias de usuários contidas no *backlog* do *release* são selecionados conforme critérios definidos pelo dono do produto e equipe

de desenvolvimento para compor o *backlog* da iteração. Esse *backlog* é consumido para produzir o incremento de *software* da iteração. Durante a execução da Iteração, a critério da contratada, são executadas reuniões diárias e, ao final da iteração, é realizada a demonstração do produto e a retrospectiva da iteração. A validação do incremento de *software* da iteração poderá ocorrer em paralelo com a próxima iteração.

Figura 5. Planejamento da Iteração.

15.4. FLUXO DE VALOR NA TRANSIÇÃO

As atividades de transição estão destacadas na figura 2 pela cor roxa. Cada *release* é homologado e liberado nas atividades de transição na forma de uma versão funcional e pronto para disponibilizar no ambiente de produção. Portanto as atividades de transição visam garantir que clientes e usuários tenham condições de usar o produto de *software*.

Para facilitar o entendimento da cadeia de valor do produto de *software* apresentado nos grupos de atividades de construção do projeto apresentado na Figura 2, foi disponibilizado no Anexo II uma exemplificação dos itens que participam da cadeia de valor.

PARTE III

16. DESCRIÇÃO DE ATIVIDADES RELACIONADAS AO MODELO DE REFERÊNCIA

Esta parte descreve, de forma detalhada, atividades relacionadas ao modelo de referência adotado por este Guia. Primeiro serão descritos os grupos de atividades de planejamento, construção do *release* e transição, depois os grupos de atividades de gestão de ordem de serviço, controle e acompanhamento do projeto e gestão de ambientes de TI. Cada atividade é descrita utilizando os seguintes atributos: nome da atividade, objetivo, descrição, responsável, artefatos de entrada e saída. Em algumas atividades também são apresentadas referências complementares para melhor entendimento de técnicas descritas.

As orientações para auxiliar a definição de "Pronto", típica das práticas de métodos ágeis, estão definidas na atividade que avalia qualidade disponibilizada no grupo de gestão de ordem de serviço (Avaliar, Aceitar ou Rejeitar Serviços de OSs). Essa atividade é indicada para ser realizada no final de cada iteração ou *release*. Esses critérios auxiliam na definição de critérios de aceite e qualidade de entregas para a contratada de desenvolvimento.

16.1. ATIVIDADES DE PLANEJAMENTO

Nas atividades de planejamento hachuradas na cor azul escuro, mostradas na figura 2, são definidos a visão e o *roadmap* do produto. Estes artefatos contém um conjunto de objetivos de negócio e características-chaves do produto de *software*.

A área requisitante do projeto de *software*, representada pelo dono do produto e clientes, deve ser apoiada pela contratada de desenvolvimento, principalmente neste grupo de atividades. Ela deve fornecer a compreensão do negócio, das necessidades, dos objetivos de negócio e dos requisitos do produto de *software*.

Cada atividade de planejamento define artefatos que deverão ser produzidos pela área requisitante com o apoio da contratada de desenvolvimento (artefatos de saída). As atividades de planejamento foram descritas a partir das definições do (Scrumex, 2014) e metodologias ágeis de instituições públicas (PDS-BC Ágil, 2014), (MGDS-INEP, 2013) e (MIDAS-IPHAN, 2013).

As atividades de planejamento são:

- Construir a Visão do Produto;
- Planejar o *Roadmap*.

Atividade: Construir a Visão do Produto.

Objetivo: Construir a visão do produto onde são descritas as necessidades, expectativas, objetivos específicos de negócio e proposta de solução para o projeto.

Descrição: Nessa atividade, entende-se que existe um problema a ser resolvido ou uma oportunidade a ser aproveitada. Para atender a essa demanda, um produto será construído por meio de um projeto.

O documento de visão deve ser construído buscando responder as seguintes questões:

- Qual problema, oportunidade, benefícios e necessidades que este produto/projeto resolve ou aproveita?
- Quais são os objetivos específicos de negócio do produto (Objetivos)?
- Quais os clientes e usuários interessados na solução (Atores)?
- Como clientes e usuários poderão atingir os objetivos de negócio (Impacto)?
- Quais as características-chaves (ou macrofunções) do produto final: quais funcionalidades de negócio, performance, segurança, escalabilidade, precisam ser entregues (*Features- Entregáveis*)?
- Quais os processos de negócio envolvidos nesta solução?
- Quais tarefas e atividades do processo de negócio serão automatizadas?
- Qual o ciclo de vida das entidades do negócio?
- Quais ambientes, padrões, aplicações, terá que suportar?
- Escopo e abordagem da Solução?
- Quais são os principais riscos e restrições do projeto?
- Qual a expectativa de custos e prazos?
- Quais premissas devem ser consideradas?
- Quais os diferenciais em relação à solução atual ou outra existente?

As necessidades e expectativas do cliente, relacionadas aos objetivos de negócio, serão utilizadas para definir o conjunto de características-chaves (ou macrofunções) do produto final e estarão fortemente alinhadas. Ela é um tipo de requisito de negócio de alto nível que consiste numa declaração de um objetivo do negócio que a solução deverá trazer ao ambiente do negócio, é um estado ou condição que o negócio deve satisfazer para alcançar a sua visão. (BABOK, 2011).

Os objetivos específicos do negócio devem estar sempre no foco do projeto (agregação de valor para o cliente), pois são a chave para tomar boas decisões sobre custo, escopo e prazos, tanto no início e depois, quando as mudanças acontecem. Os objetivos de negócio devem atender simultaneamente a seis critérios (Chiavenato, 2004):

- Focado em um resultado e não em um uma atividade;
- Consistente, ou seja, alinhado coerentemente a outros objetivos e demais metas da organização;
- Específico, ou seja, bem definido;
- Mensurável, ou seja, quantificável e objetivo;
- Alcançável.

O impacto é uma declaração do cliente de como poderá satisfazer seus objetivos específicos de negócio por meio de um conjunto de ações e decisões que, por sua vez, serão contempladas pelas características-chaves do produto. O impacto garante aderência entre os objetivos específicos de negócio e as características-chaves do produto (*software* features) (Impact Mapping, 2015).

Característica-chave do produto (features) é o serviço ou macrofuncionalidade que agrega valor ao negócio do cliente, dispensando uma descrição mais detalhada. Outros sinônimos usados são: funcionalidade de negócio, requisito de negócio e feature. As características do produto podem coincidir com algum processo de negócio ou parte dele, ou agrupamento de variados processos. Ela implementa um impacto por meio dos entregáveis (deliverables) (Impact Mapping, 2015).

A abordagem da solução é a forma de tratamento que será utilizada para implementar ou sustentar um novo conjunto de capacidades. Abordagens de solução descrevem os tipos de componentes de solução que serão entregues (novos processos, um novo aplicativo de *software*, hardwares, novas contratações, treinamentos, etc.) e podem, também, descrever a metodologia que será utilizada para entregar esses componentes. (BABOK, 2011).

Notas:

- A visão do produto deve ser criada utilizando uma linguagem de fácil entendimento para o cliente, visto que este será um elo de comunicação com a equipe do projeto;
- O dono do produto deve realizar reuniões com os participantes do projeto até definir a visão do produto. Algumas técnicas que contribuem para a elaboração da visão do produto são:
  - *Product Vision Box;*
  - *Elevator Statement;*
  - *Impact Mapping;*
  - *Brainstorming;*
  - Mapeamento do Processo de Negócio;
  - Entrevistas.
  - Veja mais informações sobre as técnicas citadas acima no anexo I (Conceitos e Fundamentos).

- Uma técnica útil na definição da visão do produto é o *Impact Mapping (Impact Mapping, 2015*), o qual consiste em definir objetivos (Razão ou o porquê) associados aos atores (Quem), que por sua vez, alcançam os objetivos por meio de um impacto que se pretende ou de uma forma (Como). Por último, as entregas (o quê), que realizam os impactos de modo a satisfazer aos objetivos. As entregas consistem em produtos, serviços e processos organizacionais. Orientações da técnica:
  - Associe os objetivos específicos de negócio aos atores ou público alvo;
  - Identifique os impactos necessários para que os objetivos sejam atingidos. Os impactos podem ser resultados pretendidos ou contribuições;
  - Identifique as características-chaves do produto necessárias para implementar os impactos;

Responsável:

- Dono do Produto;

Entradas:

- PDTI / PEI;
- Guias de Apoio;
- Ordem de Serviço.

Saída:

- Documento de Visão.

Atividade: Planejar o *Roadmap*.

Objetivo: Construir o planejamento de entregas ou plano cronológico de liberação dos *releases* (versões do produto). Esse plano cronológico é chamado de *roadmap.*

Descrição: Planejar o *roadmap* significa dividir objetivos de negócio e as características-chaves ou macrofunções do produto em partes entregáveis, por ordem de prioridade. Conforme o modelo de referência deste Guia, as partes são os *releases* que, por sua vez, são construídas a partir das características-chaves do produto priorizadas e ordenadas.

As características-chaves do produto são extraídas do Documento de Visão e devem estar alinhadas com as necessidades, as expectativas do cliente e, por fim, aos objetivos específicos de negócio.

Orientações para a elaboração do plano de entregas dos diversos *releases*:

- Estabeleça um *ranking* dos objetivos e das características-chaves do produto a partir do que mais agregue valor para o que agregue menos;
- Agrupe os itens (objetivos e características-chaves) por ordem de prioridade, na quantidade compatível com a capacidade de produção do projeto (instituição e contratada) e no tempo disponível para o desenvolvimento. Exemplos de escala de prioridade: Imprescindível, importante, desejável;
- Estabeleça uma cronologia de entregas ou a periodicidade;
- Organize reuniões em que os envolvidos participem ativamente da construção do *roadmap*;
- Divulgue o *roadmap* para todos os envolvidos.

O interstício entre os *releases* é o prazo para entrega de uma e início da outra. Ao final de cada *release*, o *roadmap* poderá ser revisado e atualizado.

Por fim, o plano de entrega das versões é o registro cronológico e preordenado da entrega de cada parte da solução, devidamente aprovado pelo dono do produto e clientes.

Notas:

- Defina em conjunto com a equipe de desenvolvimento e analista de infraestrutura de TI, caso necessário, a arquitetura inicial do *software* e sua identidade visual. Neste momento, pode ser realizado um evento de *workshop* do produto para discussão dessas definições;
- Agregação de valor ao negócio: a fórmula do sucesso para os projetos ágeis (vide os princípios do manifesto ágil) é priorizar pelo valor de negócio, buscando sempre a simplicidade (na especificação, implementação e execução);
- Os *releases* não têm um prazo fixo, apenas as iterações, as quais devem sempre entregar valor ao cliente. Não se recomenda executar iterações apenas técnicas (resolver débitos técnicos, por exemplo). Pode-se até mesclar revisões técnicas, mas sempre incluindo entregas que agreguem valor ao cliente.

Responsável:

- Dono do Produto.

Entradas:

- Documento de Visão;
- Guias de Apoio.

Saídas:

- *Roadmap* do Produto.

16.2. ATIVIDADES DE CONSTRUÇÃO DO RELEASE

Neste grupo de atividades será construído um ou mais *releases*, conforme o plano cronológico definido no *roadmap*. Cada *release* poderá ser construído em uma ou mais iterações (Sprints). As atividades são executadas pela contratada de desenvolvimento de *software* em conjunto com as áreas da Instituição.

As atividades de construção do *release* estão divididas em dois subgrupos de atividades. As de planejamento do *release* e as de construção de iterações. A relação de todas as atividades de construção do *release* são:

- Atividades de Planejamento do *Release*;
  - Elaborar o *Backlog* do Produto;
  - Planejar o *Release*.
- Atividades de Construção de Iterações;
  - Reunião de Planejamento da Iteração;
  - Executar a Iteração;
  - Reunião de Demonstração da Iteração;
  - Reunião de Retrospectiva;
  - Validar Incremento de *Software*.

16.2.1. ATIVIDADES DE PLANEJAMENTO DO *RELEASE*

Planejar o *release* é determinar a entrega de uma versão do produto. A elaboração do *backlog* do produto, primeiro passo antes de definir o plano do *release*, é realizado por meio do desdobramento dos objetivos de negócio e características-chaves, estabelecidas no Documento de Visão, em requisitos de *software* (funcionais e não funcionais) e em tarefas técnicas de produção de *software*, chamados itens de *backlog*. Posteriormente, esses itens de *backlog* são priorizados, estimados e separados em iterações na atividade de definição do plano do *release*.

O *release* é liberado na forma de uma versão funcional e pronta para homologar ou para disponibilizar no ambiente de produção.

Atividade: Elaborar o *Backlog* do Produto.

Objetivo: Construir e disponibilizar o *backlog* do produto, que é a lista priorizada dos itens necessários para o desenvolvimento e entrega do produto de *software*.

Descrição: O *Backlog* do Produto representa tudo que é necessário para desenvolver e lançar um produto de valor agregado ao negócio. É uma lista de todos os requisitos (funcionais e não funcionais), funções, tecnologias, melhorias e correções de defeitos que constituem as mudanças que serão efetuadas no produto para versões futuras (Guia do *Scrum*, 2013).

Veja mais informações em:

- http://guide.agilealliance.org/guide/backlog.html

Também é importante a utilização ou definição de uma arquitetura de sistema adequada ao projeto. Caso a arquitetura não esteja definida, no início do projeto, deverão ser os primeiros requisitos técnicos a serem tratados. Recomenda-se que a arquitetura inicial e identidade visual sejam definidas ainda nas atividades do grupo de planejamento.

Caso sejam necessárias alterações nos padrões de arquitetura e identidade visual para o projeto, eles devem ser incluídos no *backlog* do produto. Essas mudanças, após implementadas, devem ser refletidas nos guias específicos.

Deve ser utilizado uma escala para priorizar os requisitos, conforme mostrado na figura 6. Nesta figura também é ilustrado um exemplo de artefato *backlog* do produto.

Figura 6. Quadro do *Backlog* do Produto

O *backlog* do produto é definido pelo dono do produto e conta com a participação da equipe da contratada para identificar novos itens de *backlog* e auxiliar no registro e especificação dos requisitos do *backlog*.

O dono do produto revisa o *backlog* e prioriza os seus itens. Cada novo item é priorizado, inserido, ou removido do *backlog* pelo dono do produto, mediante os objetivos de negócio.

A lista de itens do backlog do produto será utilizada pela equipe da contratada para implementar o produto através de cada iteração.

O Documento de Visão deverá estar sempre em evidência, pois esse é o objeto de aceite e satisfação ao fim do projeto.

Ao final desta atividade deve ser produzida uma estimativa de tamanho do backlog do produto. A estimativa poderá ser feita por meio de técnicas, tais como story point ou ideal day.

Veja mais informações em:

- http://guide.agilealliance.org/guide/relative.html
- http://guide.agilealliance.org/guide/poker
- http://guide.agilealliance.org/guide/points-estimates-in.html

Os itens de backlog prioritários especificados por meio de histórias de usuário deverão atender aos critérios específicos de "História Preparada" atendendo aos critérios do INVEST (Wake, 2003):

- Certifique se o ator ou sujeito, ação e finalidade foram claramente identificados;
- Certifique se os critérios de aceitação da história de usuário foram definidos e negociados;
- Certifique se a história de usuário é simples e concisa;
- Certifique se a história de usuário é especifica ou independente, e não necessita ser decomposta em partes mais específicas;
- Certifique se a história, de fato, gera valor ao usuário;
- Certifique se a história é estimável;
- Certifique se os testes para a história de usuário foram definidos.

Veja mais informações em:

- http://guide.agilealliance.org/guide/invest.html
- http://guide.agilealliance.org/guide/user-stories.html

O *backlog* do produto é dinâmico no sentido de que ele está constantemente sendo priorizado e mudando para identificar o que o produto precisa para ser útil, ou seja, agregar valor ao cliente (alcançar os objetivos de negócio). Esse refinamento é conhecido por *backlog grooming* e gera as histórias preparadas para serem construídas nas próximas iterações a iniciar.

Vejam mais informações em:

- http://guide.agilealliance.org/guide/backlog-grooming.html

Durante o *grooming*, o Documento de Visão deverá ser analisado com o intuito de alinhar o andamento do projeto. Artefatos de planejamento poderão ser refinados e atualizados de acordo com eventuais mudanças de negócio no projeto.

Nota:

- Nas atividades de planejamento do produto caso seja identificado um grande conhecimento da solução a ser produzida (objetivos de negócio, metas e características-chaves), com pequenas possibilidades de mudanças, a elaboração do *backlog* do produto, poderá ser elaborado logo após a concepção do Documento de Visão. Essa estratégia possibilita que o planejamento do *roadmap*, com o *plano cronológico de entrega dos releases* seja estimado com maior precisão.

Responsável:

- Dono do Produto.

Entradas:

- Documento de Visão;
- Guias de Apoio;
- *Roadmap* do Produto.

Saídas:

- *Backlog* do Produto.

Atividade: Planejar o Release.

Objetivo: Definir o plano do release com a meta a ser alcançada em função dos objetivos de negócio e características-chaves do produto.

Descrição: Nesta atividade é importante estabelecer uma meta para o *release* em função dos objetivos de negócio e do prazo.

O planejamento do *release* parte do pressuposto que o *backlog* do *release* já foi definido a partir dos objetivos de negócio e características-chaves do *release*. Uma das primeiras atividades para elaborar o plano do *release* é realizar a estimativa de tamanho e esforço, utilizando técnicas de estimativa de práticas ágeis, para implementar os itens do *backlog* do produto selecionados para o *release* (*Backlog* do *Release*). Com a estimativa de tamanho e esforço definida, é possível estimar a duração e a quantidade de iterações do *release*.

A fragmentação do *release* em iterações (de duração fixa) ocorrerá conforme a priorização e complexidade dos itens do *backlog*, o tempo disponível para a construção do *release* ou do projeto, estimativa de tamanho/esforço dos itens de *backlog*, histórico de produtividade da instituição e a estratégia de desenvolvimento estabelecida para o produto.

As estimativas dos requisitos de maior prioridade, que já foram especificados, são mais precisas, enquanto as estimativas dos itens de menor prioridade apresentam maior probabilidade de variação.

Apesar de existirem várias alternativas possíveis, deve-se procurar definir a quantidade de Iterações no plano do *release*. É fortemente recomendável adotar durações iguais para as Iterações. Isso ajuda a manter um ritmo constante de desenvolvimento e entrega pela equipe do projeto.

Veja mais informações em:

- http://guide.agilealliance.org/guide/frequent-release.html

Também podem ser inseridos no plano do *release* premissas, impedimentos e riscos envolvidos no *release*, além de prever atividades prévias ao início das iterações para que a equipe execute a criação/disponibilização dos ambientes de desenvolvimento e de testes necessários.

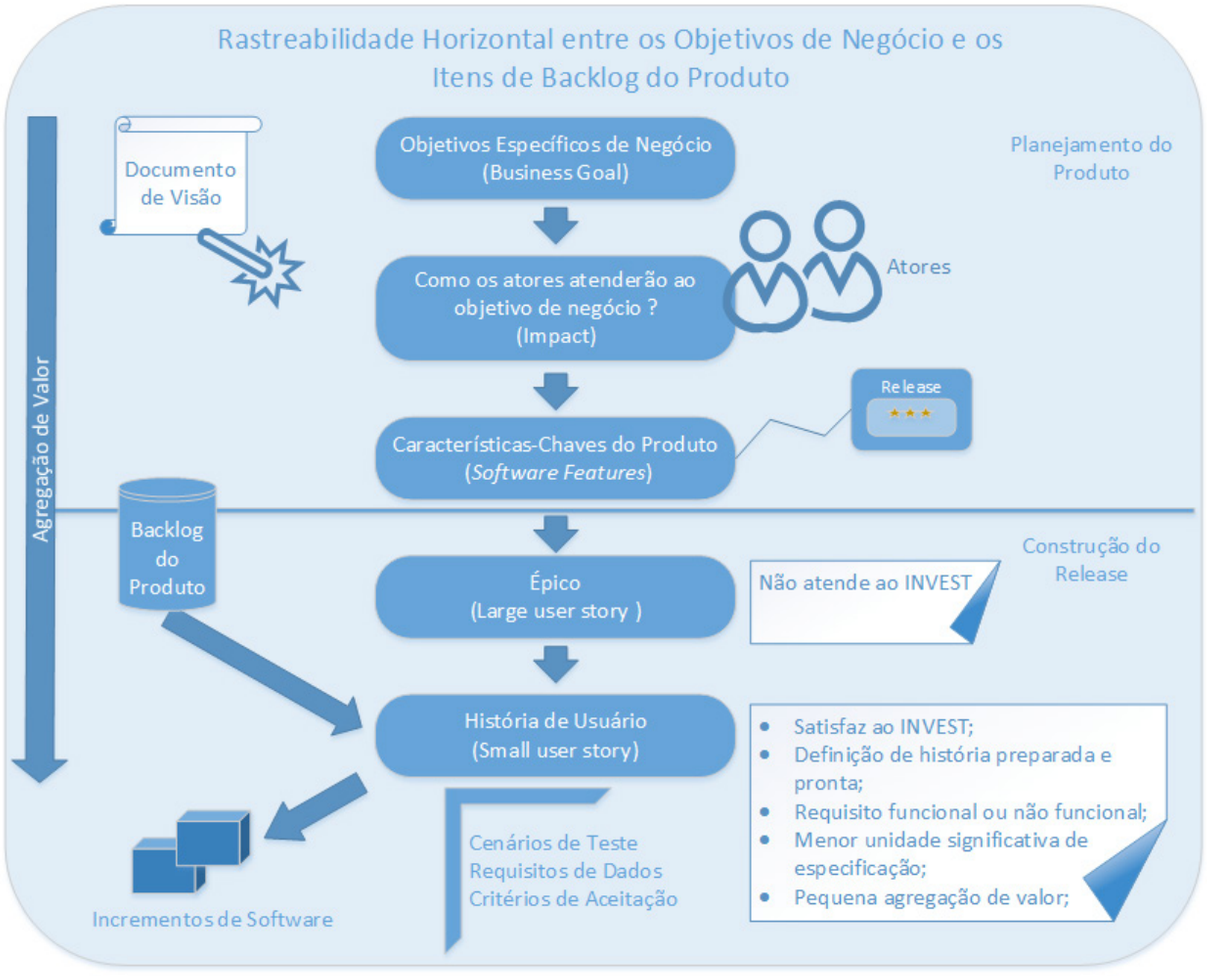

Figura 7. Rastreabilidade entre Objetivos de Negócio e Itens de *Backlog*.

Nesta atividade é importante definir os conceitos e critérios de história pronta e preparada. Também é importante aumentar o escopo do conceito de "Pronto", como por exemplo: o incremento de produto gerado pela iteração será entregue testado, integrado aos demais sistemas de informação, documentado conforme os padrões da instituição e com os usuários finais treinados. Dependendo da complexidade do incremento de *software*, pode ser definido um conceito de "Pronto" para cada item do *backlog*.

Veja mais informações em:

- http://guide.agilealliance.org/guide/definition-of-done.html
- http://guide.agilealliance.org/guide/definition-of-ready.html

Por fim as informações do plano do *release* podem ser inseridas em gráfico de *burndown*, para o seu devido acompanhamento durante o projeto.

Nota:

- Determinar um tempo fixo para a iteração. O *release* não possui um tempo fixo, pois o seu tempo depende das histórias e suas estimativas de esforço, para alcançar os objetivos de negócio. Prazo do projeto é fixo.

Responsável:

- Dono do Produto.

Entradas:

- Documento de Visão;
- *Roadmap* do Produto;
- *Backlog* do Produto.

Saída:

- Plano do *Release*.

16.2.2. ATIVIDADES DE CONSTRUÇÃO DE ITERAÇÕES

Com o plano do *release* estabelecido e atividades prévias ao desenvolvimento realizadas, é possível dar início à execução da primeira iteração e posteriormente das outras. A iteração é uma das principais atividades do projeto, afinal, é por meio desses ciclos de trabalho que são transformados os itens de *backlog* em um incremento de *software* funcional.

Atividade: Reunião de Planejamento da Iteração

Objetivo: A finalidade principal da reunião é realizar o planejamento da Iteração a ser executada. A reunião deve ser rápida e objetiva, buscando não comprometer mais de 4 horas de duração.

Descrição: Iterações são ciclos de execução do projeto, com duração fixa de 2 a 4 semanas, que tem por finalidade a entrega de um incremento do *software*.

Antes do início desta atividade, os requisitos e as histórias de usuário priorizadas para a iteração deverão atender aos critérios específicos de "História Preparada" definidos no planejamento do *release*.

Tarefas gerais da primeira parte da reunião:

- Selecionar o *backlog* da iteração a partir do *backlog* do *release*: O dono do produto, junto com a equipe da contratada, seleciona os itens prioritários do *backlog* que serão trabalhados na iteração.
- O dono do produto pode realizar alterações de inclusão, exclusão ou alteração do backlog nesta reunião. Alterações no backlog da iteração não são permitidas durante a execução da iteração, a menos que acordadas entre o dono do produto e a equipe de desenvolvimento, mediante troca de prioridades.
- Conforme (Scrumex, 2014), são fatores que influenciam na seleção dos itens do backlog:
  - Prioridade/Valor;
  - Tamanho/Estimativa;
  - Duração da Iteração;
  - Restrições/Riscos;
  - Investimento/Custo;
  - Significado*/Empacotamento.

Para o item Significado*/Empacotamento, os itens devem ser agrupados por afinidade e relação semântica de forma que agreguem valor ao negócio do cliente. Verifique o sentido que faz a implementação de um conjunto desses itens por afinidade.

Tarefas da segunda parte da reunião:

Na segunda parte da reunião, a equipe da contratada trata a questão do "como?", ou seja, decide como transformará os itens selecionados do *backlog*, que foi definido na parte anterior da reunião, em um incremento de *software* funcional. Usualmente, a equipe de desenvolvedores começa projetando o trabalho e identificando tarefas, que são peças detalhadas do trabalho necessário para converter o *backlog* em *software* funcional. As tarefas definidas também devem ser estimadas e sequenciadas. Não é necessário definir tarefas de todos os itens do *backlog*. Por exemplo, elaborar inicialmente as tarefas dos itens mais prioritários.

As tarefas definidas podem ser inseridas em um quadro de tarefas, para o seu devido acompanhamento pela equipe de desenvolvimento.

O dono do produto está presente nesta parte da reunião para esclarecer dúvidas sobre os itens do *backlog* e para ajudar a efetuar mudanças, caso estas sejam necessárias.

O Plano da Iteração deve ser elaborado, e deve conter informações da iteração, tais como: meta de negócio da iteração acordada entre as partes, data de início e fim da iteração. O *backlog* da iteração deve ser rastreável por este plano.

Outras pessoas podem ser convocadas para a reunião (ex: analista de infraestrutura, líder de projetos, administrador de dados, analista de métricas, analista de qualidade) para fornecer conselho técnico ou específico do domínio em questão.

A partir deste ponto, o objetivo é o alcance da meta estabelecida para a iteração, através da entrega do incremento de produto, conforme conceito de "Pronto" acordado com o dono do produto.

Notas:

- O prazo de entrega corresponde ao período de tempo que a contratada tem para executar a iteração, realizar os testes funcionais e/ou de requisitos não funcionais e gerar a *build* potencialmente implantável. O ideal é entre 2 e 4 semanas.
- É importante negociar nesta reunião uma agenda de compromissos da equipe junto ao dono do produto para o período de execução da iteração.
- O tempo da reunião pode corresponder, numericamente, ao tempo da iteração. Por exemplo, para uma iteração com 4 semanas, a reunião poderá durar 4 horas.

Responsáveis:

- Dono do Produto;
- Mestre *Scrum;*
- Equipe de Desenvolvimento.

Entradas:

- Plano do *Release*;
- *Backlog* do Produto (priorizado);
- Histórias Preparadas.

Saídas:

- Plano da Iteração;
- *Backlog* da Iteração;
- *Backlog* do Produto (atualizado);
- Agenda de Compromissos do Dono do Produto.

Atividade: Executar a Iteração.

Objetivo: Construir e entregar um incremento de *software* das funcionalidades e outros requisitos de *software,* listados nos itens do *backlog* da iteração, para inspeção pelo dono do produto. Um incremento de *software* é uma versão funcional do *software*.

Descrição: Após a escolha do *backlog* da iteração, caso necessário, a equipe da contratada em conjunto com o dono do produto iniciará a especificação complementar das histórias definidas, o que pode ser por meio de técnicas de elicitação e análise de requisitos. Vale destacar que existe uma diferença entre história preparada e pronta.

Nesta atividade deve-se evitar as alterações no *backlog* da iteração que causem impactos negativos no alcance de sua meta. Portanto, mudanças que impactem na meta devem ser registradas no *backlog* do produto ou do *release*. Mudanças que não causarem risco a meta da iteração poderão ser realizadas durante a iteração.

Em cada iteração são executadas todas as disciplinas de um projeto tradicional, ou seja, definição/especificação de requisitos, projeto, construção e teste, conforme pode ser visto na figura 8. Isso é necessário porque o objetivo da iteração é entregar uma parte do produto final para inspeção pelo cliente.

Veja mais informações em:

- http://guide.agilealliance.org/guide/iteration.html

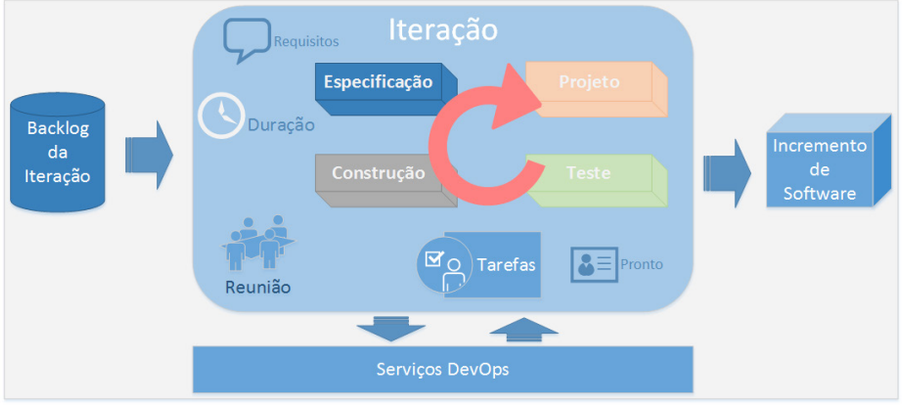

Figura 8. Elementos constituintes de uma iteração.

O mestre *Scrum* deverá facilitar o trabalho da equipe de desenvolvimento, removendo desvios e impedimentos encontrados e garantindo o bom andamento do trabalho.

Durante a iteração a equipe poderá sentir necessidade de consultar o dono do produto, o qual deverá estar disponível de forma a não comprometer os objetivos propostos.

Os itens do *backlog* da iteração deverão ser tratados por prioridade, ou seja, obedecendo as prioridades definidas no *backlog*.

Outro aspecto importante na execução da iteração é o teste em diversos níveis de cobertura. A Figura 9 apresenta os tipos de testes, em quatro quadrantes Q1, Q2, Q3 e Q4, que normalmente são realizados pela equipe de desenvolvimento e outros envolvidos, como por exemplo, os usuários especialistas no negócio.

O primeiro quadrante (Q1) lista os tipos de teste realizados por pessoal de tecnologia da informação, por exemplo, analistas de testes, testadores e analistas de requisitos. Os testes podem ser automatizados ou manuais, e oferecem suporte à equipe de desenvolvimento, pois existe uma forte interação entre os desenvolvedores e os testadores. A cobertura de testes do primeiro quadrante é orientada pelo negócio e sua execução pode ser sistemática ou pontual (Agile Testing, 2009).

O segundo quadrante (Q2) elenca os tipos de teste realizados pelos usuários e o dono do produto, tem forte orientação ao negócio, a execução é normalmente sistemática, a operação é manuale o propósito é a validação e a aceitação do produto entregue (Agile Testing, 2009).

Veja mais informações em:

- http://guide.agilealliance.org/guide/exploratory.html
- http://guide.agilealliance.org/guide/acceptance.html
- http://guide.agilealliance.org/guide/usability.html

O terceiro quadrante (Q3) elenca o tipos de testes realizados pela equipe de desenvolvimento para efetuar a verificação do código-fonte implementado ou a integração entre os componentes. Oferece suporte à equipe de desenvolvedores para facilitar e agilizar a correção dos erros e defeitos encontrados. Normalmente são testes automatizados, executados na integração contínua, mas podem ser executados em tempo de desenvolvimento (Agile Testing, 2009).

Veja mais informações em:

- http://guide.agilealliance.org/guide/unittest.html

O quarto quadrante (Q4) apresenta tipos de testes para verificar a qualidade do produto quanto à diversos requisitos não funcionais (desempenho, segurança, capacidade, robustez, etc). Estes testes podem se valer de várias ferramentas e métodos para a sua execução e podem exigir ambiente específico. Tem forte apoio da tecnologia e recaem sobre produtos de *software* em funcionamento (Agile Testing, 2009).

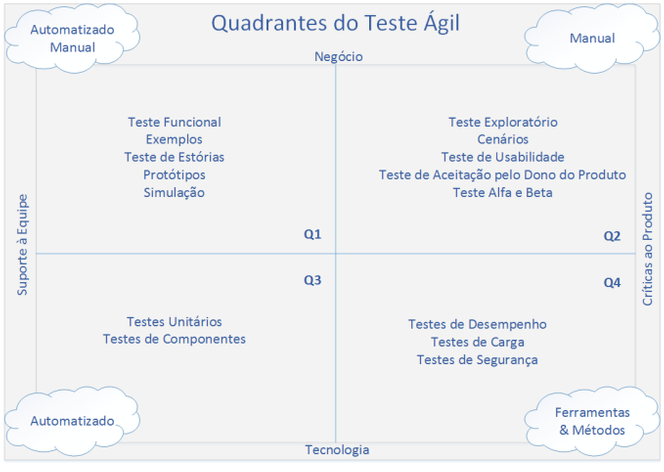

Figura 9. Quadrantes do Teste Ágil.

O suporte à equipe é resultado de uma interação sinérgica entre os testadores e os desenvolvedores para aperfeiçoar o funcionamento do *software* ou para resolver quaisquer inconformidades. As críticas ao produto significam elogios e sugestões de melhoria, oriundos de usuários especialistas no negócio ou especialistas em suporte a sistemas de informação baseados em computador (Agile Testing, 2009).

As sugestões de melhoria poderão ser convertidas em itens de *backlog* do produto de acordo com os interesses do dono do produto.

Durante a iteração, o Documento de Visão deverá estar sempre em evidência, pois esse é o objeto de aceite e satisfação ao fim do projeto. Eventualmente, poderá ser refinada.

A equipe de desenvolvimento deverá aplicar as boas práticas de engenharia de *software* para garantir a qualidade do incremento de *software* que será entregue, conforme os guias ou documentos de apoio. Algumas boas práticas sugeridas são:

- *Refactoring* (melhorar o código-fonte sem alterar comportamento);
  - Veja em: http://guide.agilealliance.org/guide/refactoring.html
- Testes unitários;
- Desenvolvimento dirigido por testes;
- Inspeção de código;
- Integração contínua;
- Padrões de projeto;
- Modularização das funcionalidades;
- Baixo acoplamento e alta coesão das funcionalidades;
- Reusabilidade de componentes.

Durante esta atividade, a contratada deverá observar as seguintes orientações:

- Elaboração e atualização de todos os artefatos/serviços previstos na ordem de serviço;
- Aplicação das boas práticas de engenharia de *software*, como a geração de código-fonte e testes unitários em conformidade com os guias disponibilizados como insumos neste processo;
- A qualquer momento, se houver necessidade de serviços nos ambientes de TI, a contratada deve registrar as demandas em sistema apropriado e dar ciência aos envolvidos no projeto, caso os serviços causem impacto na iteração;
- Realizar testes funcionais conforme o desenvolvimento das funcionalidades da Iteração (itens do *backlog* da iteração) no ambiente de testes;
- Seguir os procedimentos e *checklists* descritos nos critérios de aceitação dos produtos da ordem de serviço.

Em relação ao cancelamento da Iteração:

- Uma iteração pode ser cancelada antes do seu *time-box* terminar. O poder de cancelar uma iteração deve ser do dono do produto ou gestor do contrato. As circunstâncias do cancelamento devem ser previstas em contrato.
- Geralmente a iteração deve ser cancelada se ela não faz mais sentido para o negócio ou porque a contratada desrespeitou cláusulas contratuais, como prazo ou quantidade de entregas previstas na ordem de serviço.
- O cancelamento de iterações consome recursos, já que todos tem que se reagrupar em outra reunião de planejamento de iteração para iniciar outro ciclo.

Recomendações:

- Quanto menor o tempo da iteração, maior sua chance de sucesso, pois nas iterações longas (por exemplo, 1 mês) assuntos importantes são deixados para última hora e o objetivo pretendido corre o risco de não ser atingido.
- É necessário definir processo eficiente de gestão de requisitos junto à contratada para evitar exagero no refinamento de requisitos entre iterações;
- A equipe de desenvolvimento deverá ser auto-organizável, multidisciplinar, comprometida, comunicativa e focada no plano da iteração. Outros pontos que devem ser observados são:
  - Controlar a participação da mesma equipe em diferentes projetos;
  - Definir estratégias para minimizar impactos de possível rotatividade na equipe.
- As histórias prontas e seus critérios de aceitação poderão incluir todos os artefatos de entrega para a iteração, tais como: Modelo de Dados e *Scripts*, Documento de Especificação de Requisitos, Código-fonte e Testes Unitários, Relatório de Cobertura de Testes e Evidências de Testes.

Responsável: • Mestre *Scrum*; • Equipe de Desenvolvimento.
Entradas: • Plano da Iteração; • *Backlog* da Iteração; • Guias de Apoio; • Histórias Preparadas (conforme critérios definidos); • Critérios de Aceitação dos Itens de *Backlog* da Iteração.
Saídas: • Incremento de *Software*; • Histórias Prontas (conforme critérios definidos); • Código Fonte.

Atividade: Reunião de Demonstração da Iteração

Objetivo: Reunião para apresentar requisitos e funcionalidades desenvolvidas durante a execução da iteração.

Descrição: A equipe demonstra o incremento de *software* pronto em *build* potencialmente implantável, preferencialmente em ambiente de validação, e responde a questionamentos do dono do produto.

O dono do produto avalia, em conjunto com os participantes, se o plano da iteração foi efetivamente realizado, se os critérios de aceite definidos foram atendidos, e se a meta de negócio da iteração foi cumprida. A equipe de desenvolvimento pode apresentar outros itens construídos na iteração que não necessariamente compõem a meta, e que devem ser validados pelo dono do produto.

O dono do produto discute o estado atual do *backlog* e o grupo inteiro colabora mediante o exposto e o que isso significa com relação ao que fazer nas próximas reuniões de planejamento.

Possíveis ações decorrentes desta reunião:

- Devolver ao *backlog* funcionalidades não terminadas (que foram previamente acordadas com o dono do produto durante a execução) e repriorizá-las;
- Repriorização do *backlog*;
- Solicitar o fechamento de um *release* com as funcionalidades demonstradas sozinhas ou com incremento de iterações anteriores;
- Escolher não avançar mais com o projeto ou não autorizar outra iteração;
- Reprovar a entrega, caso a mesma não tenha atingido o mínimo aceitável em relação a meta da iteração.

Conforme o Guia do *Scrum*, 2013, no planejamento adaptativo, quando um inspetor determina que a execução de um processo desviou para fora dos limites aceitáveis, e que o produto resultado será inaceitável, o material produzido deverá ser ajustado. O ajuste deve ser realizado o mais breve possível para minimizar mais desvios. Esse guia prescreve eventos, contidos dentro dos limites da iteração, para inspeção e adaptação, quais sejam:

- Planejamento da Iteração;
- Reunião de Demonstração da Iteração;
- Reunião de Retrospectiva da Iteração.

Assim sendo, o planejamento adaptativo permite a concepção de ações para resolver desvios e impedimentos identificados na execução das iterações.

Nota:

- Um ambiente de validação também pode ser utilizado, diferentemente daquele de homologação, pois com os pequenos ciclos utilizados no ágil, pode se ter uma versão de software (build) sendo validada e uma outra versão do mesmo software sendo homologada. Durante as reuniões de demonstração, pode-se utilizar a versão do software hospedada neste ambiente;

Responsáveis:

- Equipe de Desenvolvimento;
- Mestre *Scrum*.

Entradas:

- Plano da Iteração;
- *Backlog* do Produto;
- *Backlog* da Iteração;
- Gráfico de *Burndown* do *Release* (opcional);
- Incremento de *Software*.

Saídas:

- *Backlog* do Produto (atualizado);
- *Feedback* da Demonstração.

Atividade: Reunião de Retrospectiva.

Objetivo: Nesta reunião são avaliados os pontos fortes e fracos durante a execução da iteração, visando a melhoria contínua no processo de trabalho.

Descrição: Retrospectivas são ajustes naturais em um ambiente de trabalho ágil. Ela ajuda as pessoas a melhorar as práticas e tratar problemas e obstáculos.

A equipe discute o que ocorreu de bom durante a iteração e quais problemas foram enfrentados, assim como esses problemas foram resolvidos.

O dono do produto e líder do projeto devem participar da reunião. Após analisar e debater o trabalho realizado na iteração, deverão ser registradas as lições aprendidas e as ações de melhoria para as próximas iterações.

Segundo o guia do *Scrum*, 2013, alguns objetivos se destacam nesta atividade de melhoria contínua:

- Inspecionar como a última iteração foi em relação às pessoas, aos relacionamentos, aos processos e às ferramentas;
- Identificar e ordenar os principais itens que foram bem e as potenciais melhorias; e,
- Criar um plano para implementar melhorias no modo que o dono do produto, o mestre *Scrum* e a equipe de desenvolvimento fazem seu trabalho.

Finalmente, a melhoria contínua do processo é importante para aumentar a qualidade da execução e, consequentemente, a garantia da qualidade do produto entregue.

Recomendação:

- Para facilitar a gestão de tempo e evitar reuniões com assuntos repetitivos as reuniões de demonstração e retrospectiva poderão ser agrupadas em uma única reunião;
- Analisar junto ao mestre *Scrum* a performance da equipe na iteração.

Responsáveis:

- Mestre *Scrum;*
- Equipe de Desenvolvimento.

Entradas:

- Ações de Melhoria das Iterações Anteriores;
- Relatório de Desvios e Impedimentos, Dificuldades ou Problemas Encontrados.

Saídas:

- Lições Aprendidas;
- Ações de Melhoria.

Atividade: Validar Incremento de *Software*
Objetivo: A validação é a ratificação dada pelo dono do produto e clientes aos artefatos e incremento de *software* entregues pela contratada. A validação também é importante para que se produzam efeitos administrativos, como os de faturamento. **Importante:** A validação do incremento de *software* da Iteração recém-finalizada (N) deve ocorrer após o planejamento e em paralelo à construção da próxima iteração (N+1), de forma a não interromper o ritmo da equipe de desenvolvimento. Essa atividade deve ter um tempo fixo previamente acordado e que não exceda a duração da iteração em curso (N+1).
Descrição: A validação do incremento de *software* da iteração deve ser realizada após o término do trabalho. É importante que a validação aconteça, preferencialmente, no ambiente de validação disponibilizado pela infraestrutura, quando este existir. Os testes ilustrados na figura 9, no quadrante 2, são exemplos de testes que serão realizados pelo dono do produto e usuários durante a validação. Caso a validação seja aceita, o dono do produto dará ciência à área de TI. Os itens identificados durante essa atividade,sejam mudanças de requisitos ou não conformidades, não devem impactar ou alterar o *backlog* da iteração em curso (N+1) e sim comporem o *backlog* do produto para serem priorizados pelo Dono do Produto conforme sua necessidade. Caso sejam encontradas não conformidades com os requisitos originais, o cliente dará ciência ao dono do produto. Esse valida, registra e encaminha a lista de não conformidades à contratada e à área de TI. No anexo IV estão listados exemplos de não conformidades. **Recomendações:** • Validações pontuais podem ocorrer na medida em que as histórias de usuário são implementadas e disponibilizadas, desde que não causem impactos nas entregas da equipe de desenvolvimento; • Se qualquer resultado ou incremento de *software* for reprovado pela validação do dono do produto, baseado nos requisitos de *software*, será ônus da contratada providenciar a correção e a nova entrega. Podendo ocorrer também a inclusão de outras penalidades administrativas pela violação de qualquer regra contratual ou Contrato de Nível de Serviço.
Responsável: • Dono do Produto.
Entradas: • Histórias de Usuário ou Especificação complementar de requisitos, conforme Guia de Especificação de Requisitos; • Histórias de Usuário Prontas; • Incremento de *Software* (ambiente de homologação).
Saídas: • Relatório de Validação; • Termo de Aceite Negocial.

16.3. ATIVIDADES DE TRANSIÇÃO DO PROJETO

Este é um grupo de atividades que serve para realizar algumas verificações finais do *release* ou do projeto produzido, e para garantir uma versão útil do produto em ambiente de produção. A atividade de "Implantar *Software*", descrita no grupo de atividades de gestão de ambientes de TI é utilizada para disponibilizar um *release* do *software* em ambiente de produção.

Um novo ciclo do projeto pode ser iniciado em paralelo com esta atividade de transição, deste que tenha equipe suficiente (instituição e contratada) e que sejam garantidos os mecanismos para este paralelismo.

Atividades executadas neste grupo de atividades são:

- Homologar o *Release*;
- Preparar e Realizar Treinamentos;
- Reunião de Encerramento do *Release ou do* Projeto.

Atividade: Homologar o *Release.*
Objetivo: A homologação do *release* é a ratificação dada pelo dono do produto e clientes a integração entre todos os módulos, componentes e incrementos de *software* entregues pela contratada. A homologação também é importante para que se produzam efeitos administrativos, como os de faturamento.
Descrição: O *release* normalmente consiste em uma automação de parte do processo de negócio do cliente, podendo contemplar uma série de requisitos funcionais e não funcionais que relacionados agregam valor ao negócio. A homologação do *release* corresponde a aceitação do produto de *software* em termos funcionais e negociais, ou seja, o cliente aceita que todas as funcionalidades estão alinhadas com os objetivos de negócio e funções de negócio. É importante que a homologação aconteça sempre no ambiente de homologação disponibilizado pela infraestrutura. Caso sejam encontradas não conformidades com os requisitos originais, o dono do produto valida, registra e encaminha a lista de não conformidades à contratada e à área de TI. **Nota:** • Casos de demora de realização de homologação pelo dono do produto e usuários, após o período definido, podem ser tratados com regras específicas para não atrapalhar o cronograma de Iterações ou *Releases*.
Responsável: • Dono do Produto.
Entradas: • Histórias de usuário ou Especificação complementar de requisitos, conforme Guia de Especificação de Requisitos; • Definições de Pronto do *Release*; • *Release* do *Software*.
Saídas: • Relatório de Homologação; • Termo de Aceite Negocial.

Atividade: Preparar e Realizar Treinamentos

Objetivo: Esta atividade compreende o planejamento, preparação do material e repasse de todo o conhecimento do sistema ou *release* desenvolvido para as equipes de suporte, sustentação, gestores de negócio e usuários. Esta atividade pode envolver toda a equipe do projeto.

Orientações para o treinamento e repasses:

- O dono do produto deve planejar a estratégia de transferência de conhecimento para gestores de negócio e usuários do produto de *software*;
- O dono do produto deve, com o apoio da equipe de desenvolvimento, produzir o manual de usuário e o material de treinamento, quando necessário;
- O dono do produto deve produzir o material específico para treinamento dos gestores de negócio ou usuários;
- O líder do projeto, junto com a equipe de desenvolvimento deve planejar a estratégia de transferência de conhecimento para equipes de suporte e sustentação;
- As equipes de suporte e sustentação devem ser treinadas, pela equipe de desenvolvimento, para receber e tratar os suportes de 1o, 2o e 3o nível da solução. Esse treinamento é gerenciado pelo líder de projetos;
- Devem ser repassados, pela equipe de desenvolvimento, todos os documentos necessários ao suporte e manutenção da solução;
- Deve ser dado apoio, pelo líder do projeto, para atualização ou inclusão de informações da solução no catálogo de serviços de TI;
- Adotar formas e recursos necessários para a transferência efetiva de conhecimento entre as equipes, desde que não contrariem as normas da instituição e contratos envolvidos.

Notas:

- A sala de aula com instrutor pode ser um exemplo de recurso, para a transferência de conhecimentos;
- Há alguns casos em que o treinamento pode ser fornecido, mediante contratação específica.

Responsáveis:

- Dono do Produto;
- Líder de Projeto.

Entradas:

- *Scripts* de Atendimento ao usuário;
- Manuais ou Vídeos de Treinamento;
- Versão Funcional do Produto (ambiente de treinamento).

Saída:

- Relatório de Treinamento;
- Avaliação do Treinamento.

Atividade: Reunião de Encerramento do *Release* / Projeto
Objetivo: Realizar verificações para garantir que o *release* ou produto do projeto produzido está apto para ser utilizado em produção, e que o *release* ou projeto pode ser encerrado.

Orientações para esta atividade:
• Analisar e definir ações para o tratamento dos defeitos residuais do *release*/projeto;
• Confirmar que a solução está operacional e foi transferida para a equipe de suporte e sustentação, verificando que:
  • Foram desenvolvidos todo o material de suporte (manuais, vídeos e *scripts* de atendimento);
  • Equipes de suporte (1o, 2o e 3o nível) estão preparadas para dar o suporte necessário para a continuidade da solução;
• Confirmar se os usuários foram treinados e estão aptos a utilizar a solução;
• Avaliar os critérios de aceitação da qualidade do produto;
• Aprovar os resultados do *release / projeto*;
• Caso seja o último *release* do projeto, aprovar o encerramento do projeto.
 	
O líder do projeto deve tomar algumas ações para o encerramento efetivo do projeto, tais como:
• Atualizar o status do portfólio de projetos como encerrado, caso não haja mais *releases* do projeto;
• Verificar a inclusão do novo sistema no catálogo de serviços da TI: isto facilita o gerenciamento dos serviços, tanto para a TI como para a área de negócio no momento de solicitar a evolução ou correção do sistema.

Responsáveis: • Líder de Projeto.
Entradas: • Documento de Visão; • *Roadmap* do Produto; • Planos dos *Releases*; • Relatório com Lições Aprendidas; • Relatório de Treinamentos; • Relatório de Homologação dos *Releases.*
Saídas: • Termo de Encerramento de Projeto.

16.4. ATIVIDADES DE GESTÃO DE AMBIENTES DE TI

Os ambientes de TI são utilizados principalmente para as atividades de construção da Iteração/*release*.

As descrições das atividades de gestão de ambientes de TI foram definidas com base nas experiências de especialistas na sustentação de infraestrutura de TI e em práticas definidas em projetos ágeis, tais como o *Scaled Agile Framework* (Scaled Agile Framework, 2014).

As atividades de gestão de ambientes de TI são:

- Preparar e Manter Ambientes de TI;
- Implantar *Software*.

Atividade: Preparar e Manter Ambientes de TI

Objetivo: Preparar e manter disponível os ambientes de desenvolvimento, testes funcionais, testes de requisitos não funcionais, homologação, treinamento e produção.

Descrição: No processo de desenvolvimento de um *software* vários ambientes de TI são utilizados, e cada um deles corresponde ao estágio de produção do *software*, os quais são: desenvolvimento, testes, homologação, treinamento e produção.

Esses ambientes podem disponibilizar técnicas de integração contínua para várias tarefas executadas no projeto. A integração contínua é o processo do ciclo de desenvolvimento de *software* responsável pela coleta e empacotamento dos diversos componentes integrantes da aplicação, assim como a sua disponibilização em ambiente previamente determinado. As equipes utilizam-na para integrar rapidamente as suas implementações numa versão ou incremento de *software*. Alguns de seus benefícios são:

- Correção de erros mais rápida;
- Testes e retestes mais eficazes;
- Maior controle sobre as ações de disponibilização.

Veja mais informações em:

- http://guide.agilealliance.org/guide/cd.html

A gestão de configuração e mudanças é um processo de engenharia de *software* relacionado a esta atividade, pois é capaz de garantir o histórico de mudanças nos artefatos, facilitando a recuperação e a responsabilização dos artefatos.

Veja mais informações em:

- http://guide.agilealliance.org/guide/versioncontrol.html

Os *DevOps* disponibilizam serviços e aplicações diversos para apoiar as atividades de engenharia de *software*, gestão de projetos, gestão de contratos e gestão de serviços. Os *DevOps* viabilizam a operacionalização das soluções. Os processos operacionais da produção do *software* podem ser automatizados e disponibilizados por meio do *DevOps*.

Diretrizes para a organização dos ambientes de TI:

- Isolamento dos estágios de construção em: testes funcionais, homologação e produção, testes de requisitos não funcionais e treinamento;
- Versionamento;
- Mecanismos de backup;
- Centralização dos artefatos;
- Automação da integração contínua;
- Interoperabilidade;
- Rastreabilidade das modificações;
- Gestão de configuração e mudança;
- Automação da cobertura de testes;
- Reportabilidade automatizada de erros e defeitos;
- Controle de qualidade do *software*.

Veja mais informações em:

- http://guide.agilealliance.org/subway.html

Notas:

- Um ambiente de validação também pode ser utilizado, diferentemente daquele de homologação, pois com os pequenos ciclos utilizados no ágil, pode se ter uma versão de *software* (*build*) sendo validada e outra versão do mesmo *software* sendo homologada. Durante as reuniões de demonstração, pode-se utilizar a versão do *software* hospedada neste ambiente;
- O Ambiente de testes de requisitos não funcionais é outro ambiente de testes que também pode ser previsto, para utilização em algumas situações em que este tipo de teste (carga, desempenho, segurança, etc) se mostrar como requisito relevante para o projeto;
- Caso seja contratado o serviço de infraestrutura de TI, os ambientes serão fornecidos pela contratada. Nesse caso, o Analista de infraestrutura de TI também deverá pertencer à contratada.

Responsável:

- Analista de Infraestrutura de TI.

Entradas:

- Guias de Apoio (Arquitetura, Interface, Configuração, Mudança e outros);
- Documento de Visão;
- Código-fonte;
- *Scripts* de banco de dados;
- Procedimentos de Implantação;
- Requisitos dos Ambientes.

Saídas:

- Ambientes de TI (Desenvolvimento, Testes, Homologação, Treinamento e Produção);
- Documentação dos Ambientes de TI.

Atividade: Implantar *Software*.

Objetivo: É a disponibilização do produto de *software* no ambiente alvo, de acordo com o estágio em que se encontra (homologação, testes, treinamento ou produção).

Descrição: Para dar mais dinamicidade ao processo, sugere-se a utilização de rotinas automáticas de implantação. Cada instituição tem sua política de publicação. Em algumas é permitido que a própria equipe de desenvolvimento faça publicações em alguns ambientes (testes e homologação). A publicação no ambiente de produção é controlada pela área de infraestrutura de TI.

As publicações podem ser realizadas diariamente ou ao final de cada iteração. Publicações diárias facilitam o acompanhamento e validação pela equipe da instituição, desde que tais validações não afetem a iteração corrente.

É importante a criação de um canal de implantação da versão pronta (*build*), com rotinas automatizadas para cada ambiente disponibilizado, onde se possa fazer publicação sob demanda de forma rápida e segura.

Veja mais informações em:

- http://guide.agilealliance.org/guide/autobuild.html

Para estabelecer processo e canal de implantação efetivo, é crucial que os membros da equipe de operações de publicação da infraestrutura estejam engajados no processo. A equipe de infraestrutura consiste nos seguintes perfis: administradores de sistemas, Administradores de Bancos de Dados, engenheiros operacionais, engenheiros de rede de dados entre outros. Eles são responsáveis pela publicação da solução e por manter o ambiente de TI atualizado e livre de erros.

A Figura 10 apresenta uma visão geral dos serviços envolvidos na implantação de um *software* em ambientes de disponibilização. Os desenvolvedores gravam o código-fonte no repositório central de artefatos, por meio de um serviço de configuração e versionamento; após este registro o responsável pela integração contínua inicia o processo automatizado de testes, verificação e empacotamento de componentes de *software*, análise ciclomática do código-fonte entre outras operações; ao final desses processos automatizados, uma *build* (executável, versão implantável ou *deploy*) é gerado; esse *deploy* (ou versão implantável do *software*) é enviado ao pessoal da infraestrutura de T.I., para que seja disponibilizado nos ambientes de TI.

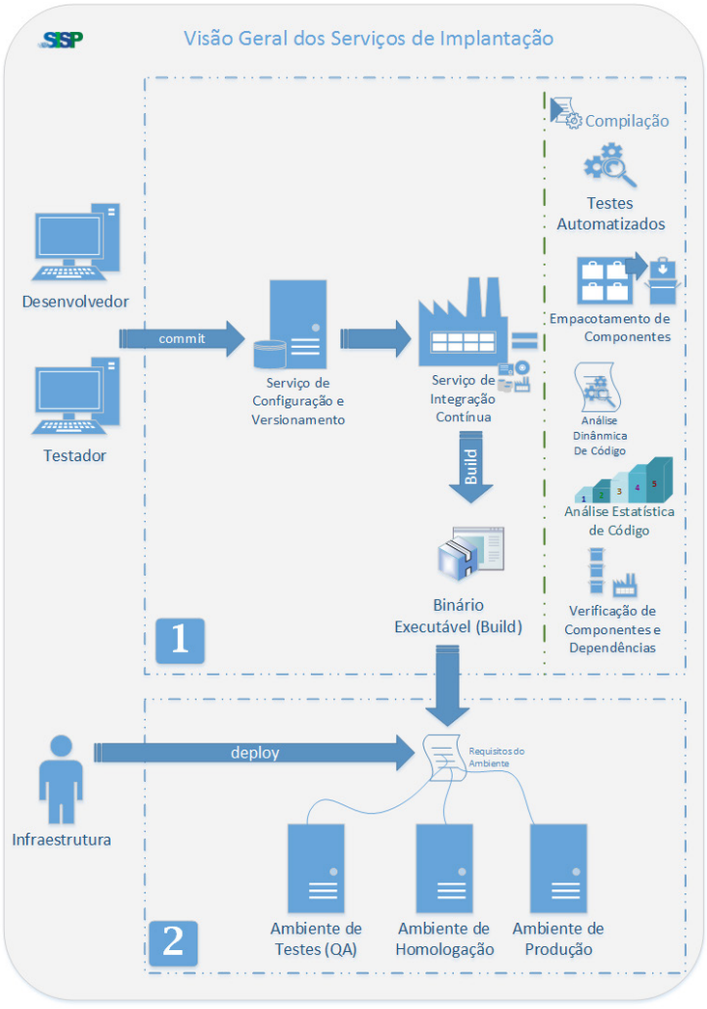

Figura 10. Visão Geral dos Serviços de Implantação.

Este modelo genérico de implantação é compatível com projetos de *software* em três camadas (*front-end* no cliente, por exemplo um *browser* HTML, *back-end* no fornecedor do serviço, um servidor de aplicação e uma camada de persistência por meio de um sistema gerenciador de banco de dados).

Notas:

- Quando contratado o serviço de infraestrutura de TI, a implantação do *software* deve ser realizada pela contratada, podendo ser feita de forma automática nos ambientes. Nesse caso, o Analista de Infraestrutura de TI pertence à contratada.

Responsável:

- Analista de Infraestrutura de TI.

Entradas: • Guias de Apoio (Arquitetura, Interface, Configuração e Mudança e outros); • Documento de Visão; • Versão Funcional do *Software* (Incremento ou *Release* de *Software*); • Procedimentos de Implantação.
Saída: • Versão *Software* Funcional no Ambiente de TI.

16.5. ATIVIDADES DE ACOMPANHAMENTO DO PROJETO

As atividades de acompanhamento do projeto devem ser realizadas durante todo o projeto. Alguns artefatos de controle e acompanhamento são preenchidos pela contratada de desenvolvimento de *software*, como por exemplo gráficos de *Burndown* e outros pelo líder do projeto. Outra atividade importante para o acompanhamento do projeto que está relacionada à capacitação da equipe é o *coaching*.

O suporte ao controle e monitoração de projetos podem ser extraídos da Metodologia de Gerenciamento de Portfólio de Projetos do SISP (MGPP-SISP, 2013) e Metodologia de Gerenciamento de Projetos do SISP (MGP-SISP, 2011).

As atividades de acompanhamento do projeto são:

- Atualizar Acompanhamento do Projeto;
- Atualizar Gráfico *Burndown*;
- Realizar Coaching.

Atividade: Atualizar Acompanhamento do Projeto.

Objetivo: Realizar esta atividade significa acompanhar cronograma e riscos (desvios e impedimentos), atualizar artefatos e comunicar informações importantes sobre o andamento do projeto.

Descrição: O acompanhamento do projeto é realizado pelo líder de projeto, designado pela instituição. Caso seja necessário ele deve marcar reuniões periódicas com o dono do produto e a equipe da contratada para avaliar o andamento do projeto.

O projeto deve ser acompanhado periodicamente para verificar se seu andamento está de acordo com os prazos definidos (OS). E caso seja identificado problemas ou desvios, providências devem ser tomadas o mais cedo possível, para evitar impactos nas entregas do projeto.

Para facilitar o controle e acompanhamento dos desvios e impedimentos do projeto, eles devem ser classificados quanto a sua responsabilidade (instituição ou contratada).

Toda vez que a contratada, em suas reuniões ou execução de tarefas, deparar com desvios e impedimentos técnicos do projeto de responsabilidade da instituição, que impactam o andamento do projeto, o dono do produto e o líder de projetos devem ser informados. O dono do produto deve buscar, com o apoio do líder do projeto, solucionar esses problemas junto aos responsáveis da instituição, seja ele o fiscal técnico do contrato, gestor da área de *software*, analista de infraestrutura ou gestor do contrato. Quanto mais rápido as providências forem tomadas, menor será o impacto no projeto.

Recomenda-se que os projetos de *software* executados pela TI sejam cadastrados no portfólio de projetos e atualizados periodicamente, conforme as regras definidas pela instituição.

Informações como os status de cada projeto (andamento, suspensos, cancelados), prazos (se atrasados ou não) e impedimentos devem ser atualizados frequentemente. Essas Informações são atualizadas no portfólio de projetos, pelo líder do projeto e utilizadas para informar aos envolvidos no projeto.

Pode ser acordado entre o líder do projeto e a contratada, a realização de reuniões de ponto de controle para acompanhamento do andamento da iteração em periodicidade previamente negociada.

O gráfico de *burndown* gerado no projeto também pode ser avaliado periodicamente para obter informações sobre o andamento do projeto, conforme acordado com a contratada.

O gráfico de *burndown* pode ser avaliado durante a reunião de demonstração (revisão) ou durante as reuniões de ponto de controle.

Responsável:

- Líder de Projeto.

Entradas:

- Ordens de Serviço;
- Documento de Visão;
- *Roadmap* do Produto;
- Plano do *Release*;
- Plano da Iteração;
- *Backlog* do produto;
- Gráficos de *Burndown*;
- Relatos do acompanhamento com a equipe.

Saídas:

- Portfólio de Projetos Atualizado (status do projeto).

Atividade: Atualizar Gráfico *Burndown*.

Objetivo: O gráfico tem como objetivo mostrar o esforço restante para a conclusão da iteração/*release*, bem como mostrar o quão próximo ou distante a equipe está de atingir a meta da iteração ou do *release*.

Descrição: O gráfico *burndown, de uso acordado com a contratada,* é útil para acompanhar o *release* e tarefas da iteração. Estes gráficos podem ser inseridos nos planos do *release* e da iteração.

Neste gráfico, tem-se uma coluna vertical, que representa a quantidade de esforço (quantidade de trabalho que ainda precisa ser realizada, normalmente *story-points*) e uma coluna horizontal, representando as unidades de tempo, como, por exemplo, as horas ou dias para finalizar uma iteração.

Conforme a figura 11, uma linha na cor verde indica o fluxo ideal de trabalho. Quando o gráfico está acima da linha verde, a equipe de desenvolvimento está distante de atingir a meta. Em cima da linha verde, a equipe de desenvolvimento está em um fluxo de trabalho ideal. Abaixo da linha verde, a equipe de desenvolvimento está superando as expectativas.

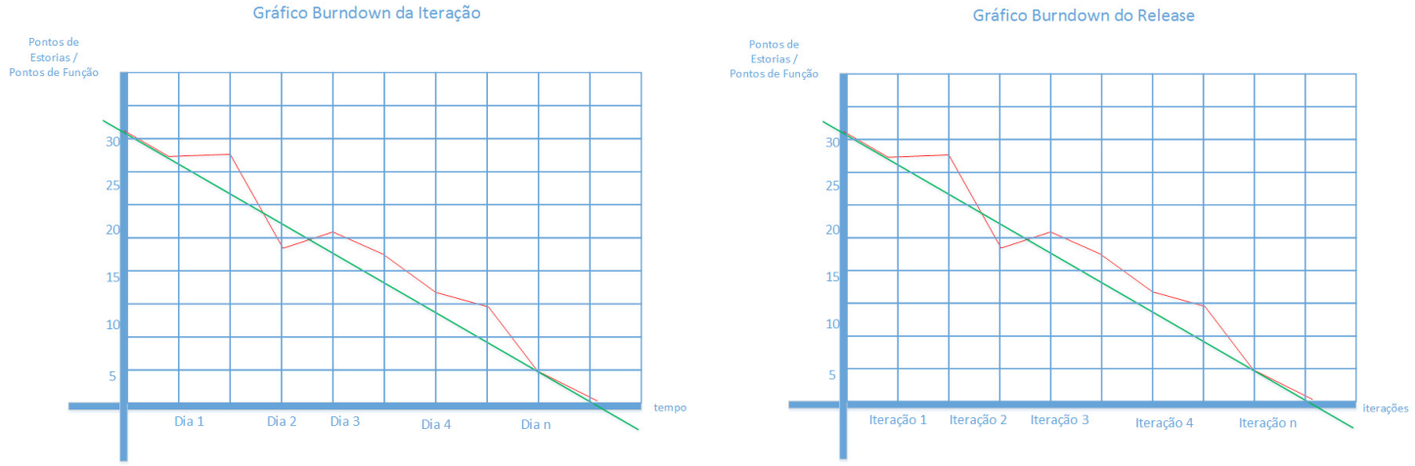

Figura 11. Gráficos de *Burndown* para *release* e iteração.

Este gráfico pode ser utilizado para avaliar o andamento do trabalho da iteração ou do *release*. Este gráfico deve ser atualizado diariamente.

Veja mais informações em:

- http://guide.agilealliance.org/guide/burndown.html

Nota:

O gráfico de *burndown* poderá ser substituído por outra técnica de acompanhamento de projetos, *release*s e iterações.

Responsáveis:

- Mestre *Scrum;*
- Equipe de desenvolvimento.

Entradas:

- *Backlog* da Iteração;
- Resultados das Reuniões Diárias;
- Plano da Iteração;
- Plano do *Release*.

Saídas:

- Gráfico de *Burndown* (atualizado).

Atividade: Realizar *Coaching*

Objetivo: *Disseminar as práticas do processo ágil na equipe.*

Descrição: Métodos Ágeis tem como objetivo primário entregar valor para o cliente. Contudo, para que a entrega de valor se concretize de fato, é necessário que a equipe do projeto tenha conhecimento adequado das suas atividades e responsabilidades.

O *coach* é um indivíduo experiente no processo ágil que zela pela sua correta execução. Ele é um mentor que trabalha lado a lado com os outros membros da equipe em suas tarefas, disseminando as práticas do processo e auxiliando a equipe a implementar software de alta qualidade alinhado às necessidades da área de negócio.

O *Coach* deve aplicar técnicas de facilitação para apoiar a equipe do projeto. Facilitação é uma abordagem que visa a oferecer meios para minimizar dificuldades, tais como: timidez, individualismo, conflitos, falta de criatividade. Essas dificuldades podem surgir durante a realização de uma determinada atividade do projeto. Seu principal objetivo é conduzir a equipe num processo de aprendizagem durante a realização das atividades do projeto.

Entre as atividades cotidianas do *Coach* ágil cita-se (Tolfo et al, 2010): a promoção e manutenção da coesão, motivação e direcionamento da equipe, a resolução de conflitos, a potencialização de novos talentos e o estímulo a enfrentar desafios.

O foco do *Coach* não é o treinamento em uma ferramenta específica, tampouco apenas resolver problemas pontuais de um determinado projeto, mas sim promover a capacidade de aprendizagem e de resolução de futuros problemas.

Diferença entre *Coach* e Mestre *Scrum*: o mestre *Scrum* está intimamente ligado a "um" processo *Scrum* para uma equipe específica, projeto e organização. Seu objetivo é amadurecer o processo para que a equipe de desenvolvimento e, indiretamente, o projeto possa ser bem sucedido. Ele facilita os eventos *Scrum* (planejamento, reuniões diárias, de avaliação e retrospectiva), e ajuda a equipe a melhorar através de inspeção e adaptação frequente. O *Coach* ágil muitas vezes trabalha com o mestre *Scrum*, dono do produto, clientes, usuários e líder de projeto, tendo portanto um universo mais amplo de atuação. O *Coach* ágil pode ajudar na transformação da equipe, introduzindo melhores práticas especialmente úteis para as empresas. Pode ajudar o líder do projeto tradicional a mudar para um papel de mestre *Scrum*. Pode auxiliar o dono do produto a entender suas responsabilidades para que possa desempenhar suas atividades. As grandes palavras que resumem a diferença são "conhecimento de transferência" e "capacitação".

Responsável: • *Coach*.
Entradas: • Práticas de Métodos Ágeis adotadas; • Atividades do Processo ágil adotado.
Saída: • Equipe capacitada para atividades do projeto.

16.6. ATIVIDADES DE GESTÃO DE ORDENS DE SERVIÇO

Conforme preconizado pela instrução normativa 04 art. 44 (In 04, 2014), o encaminhamento formal de demandas, às contratadas de TI, deverão ocorrer preferencialmente por meio de ordens de serviço (OSs).

As ordens de serviço (OSs), para esse guia, são emitidas para a contratada de desenvolvimento de *software*, ao longo de cada projeto, para apoiar e/ou executar atividades de planejamento, construção do *release* e transição, descritas no modelo de referência da figura 2.

A figura 12 descreve as fases do ciclo de vida para uma OS. Cada fase do ciclo pode estar relacionada a uma ou mais atividades de gestão de OS. A primeira fase é a de "abertura", onde é elaborada a OS. A fase seguinte é a de "execução" onde os serviços da OS são efetivamente prestados e entregues pela contratada de desenvolvimento de *software*. A fase seguinte é a de "avaliação", onde a instituição contratante faz a avaliação técnica dos serviços entregues. Assim que avaliados e aceitos os serviços da OS, o fluxo segue para a fase de "faturamento", onde é criado processo de faturamento para liquidar os serviços prestados. Assim que liquidados os valores pela área administrativa, ocorre a fase de "fechamento" da OS. A "monitoração" da execução de OS deve ser executada em paralelo ao longo de todas as fases.

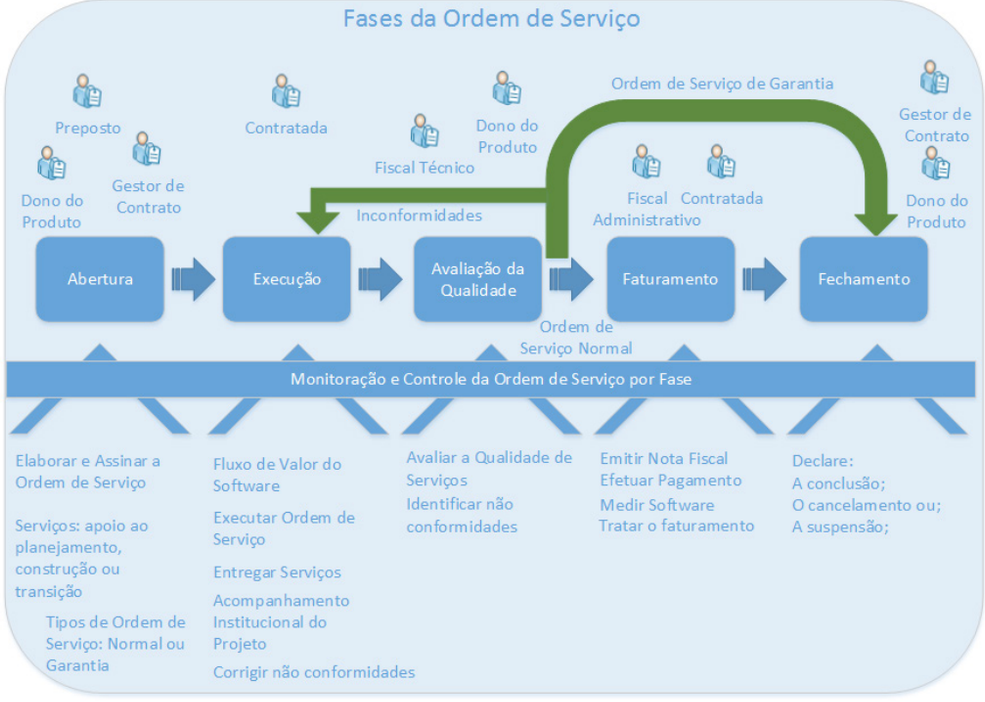

Figura 12. Fases da Ordem de Serviço.

Durante o ciclo de vida da OS, algumas atividades são de responsabilidade da instituição contratante e outras da contratada de desenvolvimento de *software*.

As principais referências utilizadas para definir as atividades de gestão de ordem de serviço deste guia foram: Instrução Normativa 04 (IN04 SLTI/MP, 2014), Guia de Boas Práticas de Contratações de TI (Guia SLTI/MP, 2014), Lei de Licitações (Lei 8666, 1993), Lei do Pregão (Lei 10.520, 2002) e Processo de Contratação de Serviços de TI para Organizações Públicas (Cruz et al, 2011). Também foram inseridas recomendações específicas nas atividades de gestão de OS, para contratos de desenvolvimento de *software* com práticas de métodos ágeis. Essas recomendações foram colhidas com gestores de instituições públicas que já praticam contratos com práticas de métodos ágeis.

É importante observar que o modelo de gestão de OS deste guia deve ser complementado por informações e atividades da legislação de contratação de TI citada acima, pois foram disponibilizadas somente atividades que descrevem orientações e recomendações importantes para os contratos de desenvolvimento de *software* com práticas de métodos ágeis.

Cada instituição poderá customizar essas atividades para se adequar aos processos e atividades já definidos.

Segue fases e atividades do modelo de gestão e fiscalização de OS proposto:

- Fase de Abertura:
  - Abrir Ordem de Serviço.
- Fase de Execução:
  - Receber Serviços de OSs;
  - Corrigir não Conformidades dos Serviços de OSs.
- Fase de Avaliação da Qualidade:
  - Avaliar, Aceitar ou Rejeitar Serviços de OSs.
- Fase de Faturamento:
  - Medir o *Software*;
  - Tratar Faturamento da Ordem de Serviço.
- Fase de Fechamento:
  - Fechar Ordem de Serviço.
- Fase de Monitoração:
  - Monitorar e Controlar Obrigações Advindas de Cláusulas Contratuais.

Atividade da Fase de Abertura:

Atividade: Abrir Ordem de Serviços.

Objetivo: Realizar a abertura de OSs para a contratada de desenvolvimento de *software*, para execução dos grupos de atividades de apoio ao planejamento, construção de *release*s e transição, conforme o modelo de referência deste guia.

Descrição: A abertura de cada OS deve preceder qualquer tarefa executada pela contratada.

Incluindo orientações da IN04, em cada OS, deve conter, no mínimo:

- A definição e a especificação dos serviços e artefatos a serem realizados;
- O Volume de serviços a serem realizados segundo as métricas definidas em contrato;
- O cronograma de realização dos serviços, incluindo atividades e tarefas significativas e seus respectivos prazos;
- Critérios de avaliação da qualidade dos serviços realizados;
- Identificação dos responsáveis pela solicitação, avaliação da qualidade e atestação dos serviços realizados, os quais não podem ter nenhum vínculo com a empresa contratada;
- Os resultados esperados dos serviços solicitados.

Recomendações para abertura de OSs:

- Caso seja aberto OS envolvendo o grupo de atividades de planejamento (construir a visão do produto e planejar o *roadmap*), a contratada deve atuar executando apenas atividades de apoio a este planejamento;
- Controlar a quantidade de OSs abertas em paralelo, de um mesmo projeto, por exemplo:
  - Não permitir a abertura de OS, para nova iteração, sem que haja um motivo justo para o paralelismo.;
  - Não permitir abertura de OS para novo *release* antes que a anterior tenha sido entregue, homologada e o *software* colocado em produção.
- Durante a execução de uma OS, caso seja identificado mudanças nas entregas definidas pelo dono do produto, deve ser gerado uma nova versão da OS baseado nas regras definidas no processo de *software* e no contrato.
- Dar preferência para a execução dos serviços no ambiente da instituição contratante, para buscar maior interação entre as equipes ágeis da contratada e da instituição e consequentemente maior qualidade dos serviços de cada OS.

Responsável:

- Gestor do Contrato.

Entradas:

- Edital da Licitação e o Contrato;
- Artefatos e Serviços propostos nas Atividades de Planejamento, Construção do *release* e Transição;
- Contrato de Níveis de Serviço.

Saídas:

- Ordem de Serviço;
- *Checklist* dos Produtos da Ordem de Serviço;
- Critérios de Aceitação e Qualidade dos Serviços da OS.

Atividade da Fase de Execução:

Atividade: Receber Serviços de OSs.
Objetivo: Para cada OS aberta pela instituição a contratada de desenvolvimento de *software* deve executar e entregar todos serviços descritos.
Descrição: A cada entrega de serviços descritos na OS, realizado pela contratada, a instituição contratante deve dar um aceite provisório, antes de iniciar a avaliação de qualidade dos produtos/serviços. **Recomendações:** • Deve ser dada atenção aos prazos de entrega dos serviços conforme cronograma definido na OS, pois o cumprimento de prazos curtos das práticas ágeis é um fator de sucesso para os projetos; • Qualquer desvio ou impedimento identificado pela contratada de responsabilidade da contratante, que inviabilize a entrega definida na OS, deve ser comunicada à instituição para que sejam tomadas as devidas providências.
Responsável: • Fiscal Técnico do Contrato.
Entradas: • Ordem de Serviço.
Saídas: • Serviços e Artefatos disponibilizados; • Termo de Aceite Provisório dos Serviços Entregues.

Atividades da Fase de Avaliação:

Atividade: Avaliar, Aceitar ou Rejeitar Serviços de OSs.
Objetivo: Avaliar a qualidade dos serviços e artefatos entregues de cada OS. Nessa avaliação podem ser gerados não conformidades conforme os *checklist*, critérios de aceitação e definições de "Pronto". Uma vez avaliados os serviços serão aceitos ou rejeitados.

Descrição: Esta atividade reúne alguns procedimentos preconizados pelo Guia de Boas Práticas de Contratações de TI (Guia SLTI/MP, 2014). Para evitar problemas de comunicação e divergências entre contratante e contratada, é importante incluir *checklists* e critérios de aceitação dos serviços de cada grupo de atividades do projeto dentro do contrato de nível de serviço ou OS.
 	
Analistas de qualidade e de infraestrutura devem apoiar a execução desta atividade.
 	
É importante a identificação e definição de critérios de "Pronto" para as entregas relacionadas a OS. Esses critérios poderão ser utilizados para avaliação da qualidade do *software* (negocial e técnica).
 	
Após avaliação técnica e negocial de uma OS, as não conformidades encontradas devem ser enviadas para a contratada efetuar a correção, não havendo mais não conformidades dos serviços e artefatos entregues, o deve ser dado o aceito definitivo dos serviços da OS.
 	
As atividades de Validar Incremento de *Software* e o *Release* são complementares a avaliação de qualidade dos serviços das OSs.
 	
Recomendações para checklists e critérios de aceitação:
 	
• Critérios gerais que podem atender a definição de "Pronto". Esses critérios facilitam a avaliação de serviços/artefatos de cada grupo de atividades:
  • Padronização dos artefatos em *templates (modelos)* definidos pela instituição;
  • Responsabilização de artefatos (autoria dos artefatos);
  • Versionamento de artefatos em repositório da instituição;
  • Ciência e concordância com o conteúdo dos artefatos.
 	
• Critérios específicos, que podem atender a definição de "Pronto" e que facilitam a avaliação dos documentos de requisitos de *software* (definição, modelagem e especificação de requisitos):
  • Adequação formal da escrita dos requisitos (funcional, não funcional, dados e regras transacionais);
  • Os requisitos são mensuráveis, em termos de função transacional e de dados;
  • Adequação material dos requisitos quanto a não ambiguidade, não redundância, ausência de função estranha, ausência de conflitos entre requisitos ou projeto, não divisibilidade (elementariedade), ausência de inconsistências, não omissão, falta de clareza e falta de coesão;
  • Os requisitos são rastreáveis até ao nível do *backlog* do produto, *release*, objetivos de negócio, características-chaves do produto, necessidades ou problemas;
  • Os requisitos são testáveis, podendo o teste ser evidenciado;
  • Os requisitos de *software* foram validados pelo dono do produto/cliente.
• Critérios específicos de "Pronto" para as entregas de Iterações ou *Releases*:
  • Itens do *backlog* da iteração ou *release* foram implementadas;
  • Evidências de teste dos itens do *backlog* da iteração foram produzidas, e os testes atingiram o padrão de qualidade esperado;

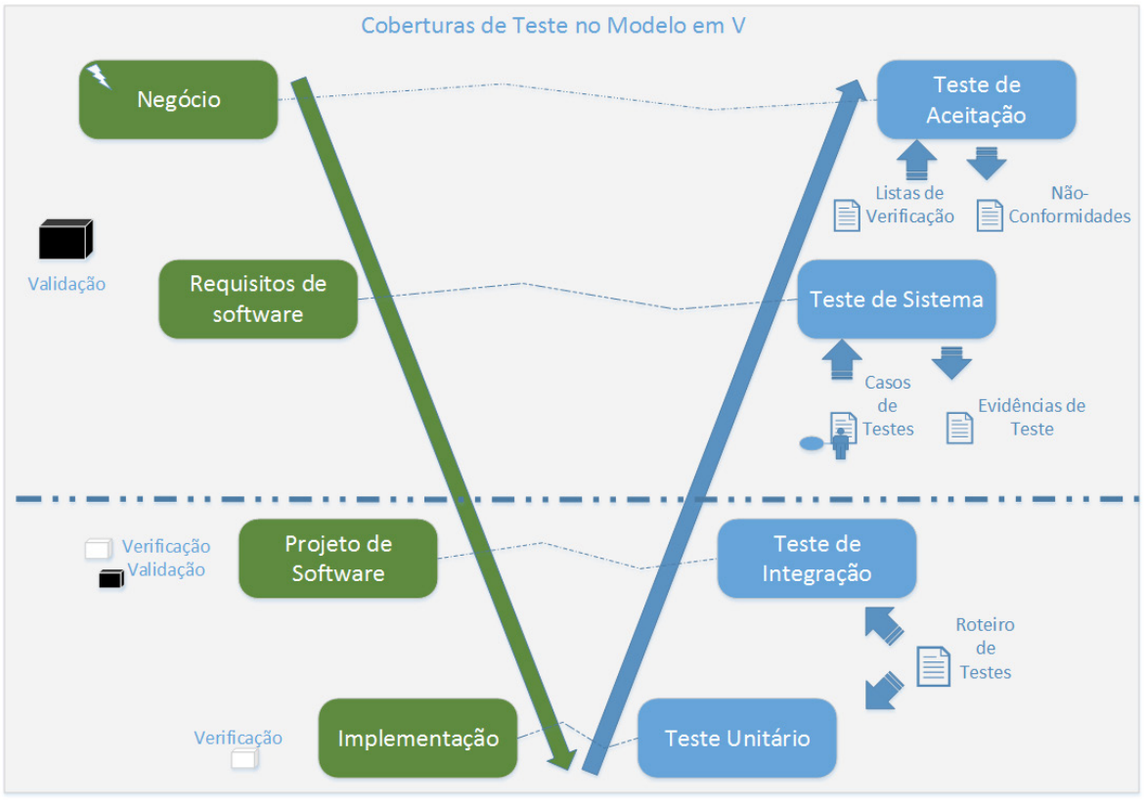

Figura 13. Modelo em V de Coberturas de Testes.

- Critérios gerais de aceitação de ordens de serviço:
  - Todos os serviços previstos na ordem de serviço foram entregues e disponibilizados no repositório de dados do projeto;
  - A qualidade da documentação entregue foi avaliada pela área de TI e as não conformidades encontradas corrigidas pela contratada;
  - A área requisitante da solução (dono do produto e cliente) aprovaram os serviços gerados e entregues;
  - A planilha com a estimativa de tamanho do projeto, *release* ou iteração foi validada pela instituição.

Outras Recomendações:

- Não dar aceite definitivo de serviços de OSs envolvendo iterações ou *releases* sem o recebimento de *software* funcional;
- Após a avaliação da qualidade, as não conformidades identificadas devem ser encaminhadas para correção pela contratada de desenvolvimento de *software*;
- É importante estabelecer o número máximo de avaliações de qualidade de uma ordem serviço que a contratada irá realizar;
- Caso a contratada não apresente um percentual mínimo de qualidade (número mínimo de não conformidades) sugere-se a aplicação de glosa no faturamento;

- Os serviços rejeitados devem ser baseados nas não conformidades levantadas. A contratada poderá apresentar justificativa para as não conformidades, que poderá ser aceita pela contratante, desde que comprovada a excepcionalidade da ocorrência, resultante exclusivamente de fatores imprevisíveis e alheios ao controle da contratada.
  - Implementação das funcionalidades estão livre de erros ou defeitos impeditivos. No anexo IV é apresentado um quadro com exemplos de não conformidades que podem ser impeditivas ou não;
  - *Software* funcional disponível no ambiente de homologação ou produção;
  - Conformidades do *software* com os guias ou documentos de apoio (arquitetura, banco de dados, interface do usuário, testes de sistemas);
  - Qualidade do código satisfaz aos indicadores pré-estabelecidos;
  - Dono do produto e clientes realizaram o aceite negocial;
  - Código-fonte foi comentado conforme os padrões de documentação.

- Avaliações técnicas de qualidade podem abranger os seguintes itens (NBRISO/IEC12119, 1998):
  - Arquitetura do *software*;
  - Interface do sistema;
  - Modelos de dados;
  - Configuração e mudança (disponibilização e gestão dos artefatos e código-fonte);
  - Realização dos testes por tipo de cobertura;
  - Artefatos previstos em cada ordem de serviço (prever *checklist* para cada artefato);
  - Código-fonte:
    - Padrões de codificação;
    - Documentação na forma de comentários;
    - Defeitos ou erros;
    - Duplicações de código;
    - Testes unitários;
    - Complexidade ciclomática (opcional).

Recomendações de Testes:A garantia da qualidade pode ser desenvolvida por meio dos testes funcionais. Conforme o modelo em V, da figura 13, os seguintes níveis de testes poderão ser contemplados: negócio, requisitos de *software*, projeto de *software* e implementação. (Rook, 1986).

Responsáveis:

- Gestor do Contrato;
- Fiscal Técnico do Contrato;
- Fiscal Requisitante do Contrato.

Entradas:

- Edital e Contrato;
- Contrato de Níveis de Serviço;
- Ordem de Serviço;
- Guias de Apoio;
- *Checklists* da OS;
- Critérios de Aceitação da OS;
- Termo de Aceite Provisório;
- Controle de Versão com Artefatos da OS.

Saídas:

- Relação de Não Conformidades;
- Rejeição dos Serviços da OS ou Termo de Aceite Definitivo;
- Relatório de Glosa.

Atividade: Corrigir não Conformidades de OSs.

Objetivo: Corrigir as não conformidades levantadas na avaliação de qualidade dos serviços entregues da OS.

Obs: Essa atividade deve ser executada pela contratada durante a realização da atividade anterior.

Descrição: Após a contratada realizar a entrega dos serviços e artefatos previstos em uma ordem de serviço, a instituição contratante realiza sua avaliação de qualidade (Técnica e Negocial). Dessa avaliação podem surgir não conformidades. Essas não conformidades devem ser repassadas a contratada para a devida correção.

Exemplos de não conformidades são:

- Defeitos ou problemas identificados no incremento de *software*, testes dos sistemas, modelo de dados, interface, arquitetura do *software*, *checklists* dos artefatos;
- Requisito de *software* não implementado;
- Qualquer tipo de débito técnico não solucionado.

No anexo IV é apresentado um quadro com sugestões de não conformidades. Para cada tipo de não conformidade são definidas recomendações de análise de impacto e consequências em termos de métricas.

Recomendações:

- Podem ser abertas OSs de garantia (sem custo para a contratante) para os serviços de correção nas conformidades;
- Estabelecer prazos para a contratada realizar a correção de não conformidades.

Responsável:

- Equipe de Desenvolvimento.

Entradas:

- Ordem de Serviço;
- Guias de Apoio;
- Relação de não Conformidades.

Saídas:

- Relação de não Conformidades (Corrigidas);
- Serviços e Artefatos Corrigidos.

Fase de Faturamento:

Atividade: Medir o *Software*.
Objetivo: Nessa atividade deverão ser realizados os serviços de medição de tamanho ou esforço do *software* para os serviços de cada OS.
Descrição: O faturamento dos serviços de cada OS deve ser realizado com base nas medições realizadas. As medições podem ser realizadas para serviços realizados em cada grupo de atividades de construção do projeto de *software*. Medição no Planejamento: No planejamento as medições de tamanho e esforço do projeto podem ser realizadas com base nos objetivos de negócio, características-chaves do produto, requisitos ou histórias definidas. Medição na Construção do *Release*: As medições podem ser realizadas no início de cada *release*, baseado no *backlog* do *release* e nas especificações dos requisitos definidos. No final do *release* deve ser realizada uma contagem com base no trabalho realizado, incluindo refinamento de requisitos, alterações e exclusões nas funcionalidades entregues e homologadas pelo dono do produto. Medição na Transição: No final de cada *release* do projeto as medições podem ser realizadas com base nas funcionalidades implementadas, homologadas e disponibilizadas em produção (valor agregado ao produto). **Recomendações:** • No anexo V é disponibilizado um quadro com itens que dão origem ao refinamento de requisitos. Esses itens devem ser refletidos na métrica de faturamento do contrato, com a definição de quem assume o ônus do refinamento de requisitos e sua forma de cálculo; • Sugere-se a criação de uma base histórica de medição para cada projeto e sistema. Essa base pode ser utilizada para dimensionamento (tamanho dos sistemas e necessidades de manutenções posteriores) e parâmetro de produtividade para novos contratos de desenvolvimento da instituição; • Medições de *software* podem ser realizadas pela contratada de desenvolvimento de *software* desde que validadas pela Instituição contratante.
Responsável: • Analista de Métricas;
Entradas: • Ordem de Serviço; • Guias de Apoio (Guia ou Roteiro de Métricas, Documento de Especificação e Modelagem de Requisitos); • Documento de Visão; • *Backlog* do Produto; • *Backlog* do *Release*; • *Backlog* da Iteração; • Código-fonte.
Saídas: • Planilha de Medição (Estimativas ou *Software* Entregue).

Atividade: Tratar Faturamento de Ordem de Serviço

Objetivo: Compor, fundamentar, apoiar e dar encaminhamentos aos processos de faturamento de OSs até seu efetivo pagamento.

Descrição: Após avaliação de qualidade, correção das não conformidades, obtenção dos aceites definitivos dos serviços, ajustados os cálculos de reduções de percebimento (glosa) previstos em contrato e edital, a contratada deve ser autorizada a emitir nota fiscal ou fatura para que a instituição faça o pagamento dos serviços prestados.

Com a nota fiscal protocolada na instituição, é criado o processo de faturamento para inclusão de todos os documentos que comprovem a prestação efetiva dos serviços descritos na OS.

O fiscal administrativo com apoio da área de contratos deve validar os valores a serem pagos em função dos quantitativos previstos e serviços prestados.

Devem ser verificados pela área administrativa (fiscal administrativo) da contratante, antes de efetuar qualquer pagamento, a regularidade fiscal, comercial, trabalhista e previdenciária. A área financeira é impedida de realizar pagamentos regulares à contratada que se encontre em situação de inadimplência para com as obrigações fiscais, comerciais, trabalhistas e previdenciárias. Portanto o fiscal administrativo deve solicitar à contratada as evidências de quitação dos encargos fiscais, trabalhistas e previdenciários. Essas evidências devem ser anexadas aos processos de faturamento.

Havendo quaisquer dessas inadimplências no momento do faturamento, o gestor do contrato deverá ser convocado para, com apoio da área de contratos, solicitar formalmente à contratada a regularização da situação, em prazo previamente definido. Havendo excedimento dos prazos concedidos, o gestor do contrato deverá tomar as sanções cabíveis.

A área financeira realiza o pagamento de OSs à contratada, baseado principalmente nos seguintes documentos:

- Contrato e ou aditivo assinado;
- Ordem de serviço assinada;
- Fatura ou nota fiscal, oriunda da contratada.
- Evidência de entrega e aceitação dos serviços prestados (Aceites);
- Planilha de Medição dos Serviços Entregues;
- Memória de cálculo dos serviços e valores a serem pagos;
- Evidência de adimplência da contratada para com as obrigações fiscais, comerciais, trabalhistas e previdenciárias.

Responsáveis:

- Gestor do Contrato;
- Fiscal Administrativo do Contrato.

Entradas:

- Edital da Licitação e Contrato;
- Contrato de Níveis de Serviço;
- Ordem de Serviço;
- Nota Fiscal ou Fatura;
- Guias de Apoio (Guia ou Roteiro de Métricas);
- Planilha de Medição dos Serviços Entregues;
- Relatório de Cálculos de Redução de Percebimento (glosa);
- Aceites Provisórios e Definitivos;
- Evidências de Quitação de Encargos Trabalhistas, Fiscais, Comerciais e Previdenciários.

Saídas:

- Documentos de Processo de Faturamento dos Serviços.

Atividade: Fechar Ordem de Serviço.

Objetivo: Finalizar OSs e realizar avaliação final dos serviços prestados.

Descrição: Após certificar de que não existe mais nenhuma pendência relativa à OS, ela poderá ser finalizada. Podem ser criados indicadores para medir o cumprimento de prazos e satisfação da área requisitante, assim como o registro de lições aprendidas.

A compilação dos resultados da avaliação podem ser utilizados para compor níveis de serviços que a contratada devera atingir.

Recomendação:

- O fechamento só deve ocorrer após certificar que os serviços foram prestados, faturados e que não existe mais nenhuma pendência relativa a OS. No fechamento da ordem de serviço além de concluída a OS pode assumir estados como cancelada e suspensa, de acordo com cada situação.

Responsáveis:

- Gestor do Contrato.

Entradas:

- Ordem de Serviço;
- Termo de Aceite ou Rejeição de Serviços;
- Não conformidades Contratuais.

Saídas:

- Ordem de Serviço (atualizada para: concluída, cancelada ou suspensa).

Fase de Monitoração:

Atividade: Monitorar e Controlar Obrigações Advindas de Cláusulas Contratuais.

Objetivo: Monitorar a execução do contrato, referente à execução de OSs, assegurando que seja executado dentro dos parâmetros previstos em edital de desempenho e qualidade (contrato de nível de serviço) e que eventuais não conformidades contratuais sejam tempestivamente tratadas de modo a evitar que o contratado incorra em ato sancionável.

Descrição: Identificadas não conformidades contratuais, deve ser gerado um registro de ocorrência, anexado ao histórico de gerenciamento do contrato e posteriormente realizar uma comunicação à contratada; (exemplos: não participação do mestre *Scrum* em atividades do projeto, equipe de desenvolvimento não formada, atrasos excessivos no projeto, serviços rejeitados, iterações canceladas por não atingir a qualidade ou meta definida). Não conformidades contratuais previstas em edital devem ser identificadas e tratadas com a notificação da empresa e a aplicação das sanções cabíveis.

As solicitações de sanções previstas no edital e no contrato, devem ser encaminhadas à autoridade competente, para aplicação de sanção instruindo processo fundamentado em caracterização exaustiva da falha cometida pela contratada e do seu enquadramento nas sanções específicas previstas.

Gerenciar riscos contratuais: Os riscos e não conformidades contratuais devem ser identificados e tratados em cada grupo de atividades do projeto. Deve-se registrar a data da verificação do risco, a probabilidade de ocorrência e os procedimentos de mitigação. Alguns exemplos de desvios são:

- Nível de serviço exigido não atendido; Mudança em processos, metodologias, especificações e/ou quantitativos definidos na contratação.

Obs: Em caso de necessidade de alteração contratual (aditivo), deve-se preservar o núcleo imutável do objeto.

Recomendação:

- O gestor do contrato deve estabelecer reuniões periódicas com a contratada para avaliar e garantir a qualidade dos serviços prestados.

Responsáveis:

- Gestor do Contrato;
- Fiscal Administrativo.

Entradas:

- Ordem de Serviço (assinada);
- Edital;
- Contrato;
- Contrato de Níveis de Serviço;
- Termos de Aceite ou Rejeição de Serviços;
- Não conformidades contratuais.

Saídas: Registro de Ocorrências; Histórico de Gerenciamento do Contrato;Notificações ou Comunicações à Contratada; Solicitação de Aplicação de Sanção.

17. BOAS PRÁTICAS PARA O SUCESSO DESTE GUIA

Para promover a entrega de projetos de *software* que cumpram as metas de valor, qualidade, prazo e custos, através do modelo de referência proposto, é importante implementar algumas ações, são elas:

- **Capacitação em Métodos Ágeis:** os métodos ágeis preconizam uma série de valores e princípios distintos do desenvolvimento tradicional de *software*. O entendimento destes valores e princípios, bem como suas práticas, são fatores de sucesso. A capacitação deve ser realizada por todos os envolvidos no projeto. O dono do produto deve entender suas funções e importância no projeto. Sua indisponibilidade para o projeto é um fator muito crítico que pode determinar o fracasso do projeto. A capacitação deve ocorrer principalmente no início dos projetos para assegurar o entendimento e envolvimento adequado dos envolvidos nos projetos;
- **Apoio ao Dono do Produto:** O dono do produto, representante da área requisitante do projeto, assume várias responsabilidades em projetos ágeis, como pode ser visto no quadro 1 papéis da contratante. Algumas instituições pesquisadas que já utilizam práticas de métodos ágeis têm utilizado meios para facilitar essa participação do dono do projeto, minimizando o risco dessa centralização. São exemplos de apoio ao dono do produto a utilização de auxiliares e ou até mesmo representantes vinculados área de TI da instituição;
- **Coaching:** diferentemente de processos de desenvolvimento de *software* mais prescritivos, baseados em documentação e na representação formal do conhecimento, os métodos ágeis são voltados para a explicitação de conhecimento implícito por meio da socialização do trabalho em equipe e interação entre pessoas. Portanto, para o sucesso do projeto e para auxiliar a equipe nas atividades do projeto é importante que a instituição contratante disponha de um profissional experiente nas práticas dos métodos ágeis adotadas, para acompanhar e treinar os envolvidos no projeto;
- **Projeto Piloto:** é importante execução de um projeto piloto, aplicando práticas de métodos ágeis, permitindo à instituição a customização do modelo de referência proposto para sua estrutura e necessidades. Essa customização, através de um projeto piloto, é fundamental para o sucesso de novos projetos. Nessa customização podem ser adequados as atividades, perfis e artefatos do modelo de referência de acordo com a estrutura da instituição (servidores e processos definidos);
- **Contratada de Desenvolvimento:** não é suficiente ter uma instituição preparada, com pessoas capacitadas e processos definidos para gestão e fiscalização de projetos de *software* com práticas de métodos ágeis. Quando se utiliza terceirização do desenvolvimento de *software,* devem ser definidos também, critérios de seleção de fornecedor que levem à seleção de empresa capacitada e com preço exequível para a prestação desses serviços de desenvolvimento;
- **Arquitetura de Referência:** utilizar arquitetura que implemente princípios de engenharia de *software*, tais como adaptabilidade ou volatilidade dos requisitos funcionais e não funcionais, flexibilidade para mudanças, alta cobertura de testes automatizados e produtividade na implementação do *software*. A implementação desses princípios facilita a entrega dos incrementos de *software* das iterações com qualidade, pois são produzidas em um curto período de tempo;
- **Padrão de Identidade Visual:** utilizar um padrão de interface visual que suporte requisitos de usabilidade, navegabilidade, funcionalidades de tela e acessibilidade, também é indispensável para criar condições de agilidade e desempenho na produtividade. Neste guia padrões de interface são representados por um guia de interface do usuário;
- **Guias de Apoio:** além dos modelos de arquitetura de referência e identidade visual, existem outras metodologias e especificações técnicas que dão suporte ao processo de desenvolvimento de projetos de *software*. Eles devem estar descritos em guias de apoio de maneira a facilitar o seu uso, tais como os guias de banco de dados, de métricas, de configuração e mudança, de segurança da informação e de testes de sistemas;
- **Modelo de Faturamento:** a remuneração da contratada de desenvolvimento de *software* depende da adoção de uma métrica de faturamento que seja objetiva e justa, dentro da legalidade e que não onere contratante nem con-

tratada. Para projetos com métodos ágeis alguns desafios são nítidos, como por exemplo, remuneração de refinamento de requisitos entre iterações. Consultar o roteiro de métricas do SISP para mais informações;

- ***DevOps*:** utilizar o DevOps, pois facilita o desenvolvimento e entrega de *software* na instituição. Eles viabilizam a construção do *software* por meio de um sistema de produção integrado de pessoas, informação, ativos de *hardware*, serviços, ferramentas e aplicativos que, por sua vez, apoiam as atividades de gestão de projetos, produção e gerência de requisitos de *software*, construção de *software*, testes automatizados, inspeção automatizada da qualidade dos produtos, integração contínua dos componentes de *software*, disponibilização automática de *releases*, gestão informatizada dos ambientes de TI, gestão de configuração dos artefatos de engenharia e gestão de mudanças. Todas estas atividades de engenharia de *software* são desenvolvidas por meio de processos, métodos e ferramentas, promovendo, assim, o princípio da impessoalidade e a diminuição do esforço da execução pela automação de processos operacionais;
- **Modelagem de Processos de Negócio:** O mapeamento de processo de negócio, realizado antes do início de projetos de *software* nas modalidades *AS-IS e TO-BE* permitem: o amadurecimento do processo de negócio; a definição de prioridades de negócio; a compreensão do negócio e suas problemáticas; a visualização e a proposição de melhorias; favorece o refinamento de requisitos; e a facilidade na transferência de conhecimento entre os envolvidos no projeto.

Outros fatores, além dos citados, podem influenciar no sucesso de projetos de *software* com práticas de métodos ágeis, visto que essa técnica pressupõe uma mudança cultural que depende de adaptações tanto de pessoas quanto de processos à legislação da administração pública. Mas experiências de bons resultados em instituições públicas, que já adotam modelo similar com métodos ágeis, devem encorajar outras instituições na sua adoção.

18. AVALIAÇÃO DE RISCOS DO ACÓRDÃO TCU 2314/2013

Em 2013, o Tribunal de Contas da União publicou o acórdão 2314 que apresenta um levantamento de auditoria sobre a utilização de métodos ágeis na Administração Pública Federal. Nesse levantamento foram apresentados alguns riscos inerentes, passíveis de materialização nas contratações com metodologias ágeis. O Anexo III deste documento apresenta uma tabela com a avaliação de cada risco apontado pelo acórdão 2314, fazendo uma correlação a este Guia.

19. REFERÊNCIAS

Agile Alliance Guide web site. Disponível em: <http://guide.agilealliance.org/subway.html>. Acesso em: maio de 2015 (Agille alliance Guide, 2015).

Agile Manifest web site. 2001. Disponível em: <http://agilemanifesto.org/iso/ptbr/>. Acesso em: junho de 2014. (Agile Manifest, 2001).

Associação Brasileira de Normas Técnicas. NBRISO/IEC12119 Tecnologia de informação - Pacotes de *software* - Teste e requisitos de qualidade. 1998. (NBRISO/IEC12119, 1998).

Beck, K. Extreme Programming Explained: Embrace Change.1 ed. Addison-Wesley. 1999. (Beck, 1999).

Brasil. Decreto-Lei N° 200, de 25 de Fevereiro de 1967. dispõe sobre a organização da Administração Federal, estabelece diretrizes para a Reforma Administrativa e dá outras providências. Portal da Presidência da República. Disponível em: http://www.planalto.gov.br/ccivil_03/decreto-lei/del0200.htm. (Decreto-Lei 200, 1967).

Brasil. SLTI do MP. Guia de Boas Práticas em Contratação de Soluções de Tecnologia da Informação Versão 2.0, Setembro de 2014, Disponível_em:<http://www.governoeletronico.gov.br/sisp-conteudo/nucleo-de-contratacoes-de-ti/modelo-de-contratacoes-normativos-e-documentos-de-referencia>. (Guia SLTI/MP, 2014).

Brasil. SLTI do MP. Instrução Normativa N° 04 de 11 de Setembro de 2014. Dispõe sobre o processo de contratação de Soluções de Tecnologia da Informação pelos órgãos integrantes do Sistema de Administração dos Recursos de Informação e Informática (SISP) do Poder Executivo Federal. Disponível_em:<http://www.governoeletronico.gov.br/sisp-conteudo/nucleo-de-contratacoes-de-ti/modelo-de-contratacoes-normativos-e-documentos-de-referencia>. (IN04 SLTI/MP, 2014).

Brasil. Ministério do Planejamento, Orçamento e Gestão. Secretaria de Logística e Tecnologia da Informação. Metodologia de Gerenciamento de Portfólio de Projetos do SISP - MGPP-SISP. Disponível em: http://www.sisp.gov.br/mgpsisp/wiki/Metodologia. – Brasília: MP, 2013. (MGPP-SISP, 2013).

Brasil. Ministério do Planejamento, Orçamento e Gestão. Secretaria de Logística e Tecnologia da Informação. Metodologia de Gerenciamento de Projetos do SISP - MGP-SISP. Disponível em: http://www.sisp.gov.br/mgpsisp/wiki/Metodologia - Brasília: MP, 2011. (MGP-SISP, 2011).

Brasil. Ministério do Planejamento, Orçamento e Gestão. Secretaria de Logística e Tecnologia da Informação. PSW- Processo de *Software* para o SISP, Versão 1.0 – Brasília: MP, 2012. (PSW-SISP, 2012).

Brasil. Banco Central do Brasil. PDS-BC Ágil - Processo Ágil de Desenvolvimento de *Software* do Banco Central, Versão 2.3.0 - Brasília: BACEN, 2014. (PDS-BC Ágil, 2014).

Brasil. Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira. MGDS - Metodologia de Gestão e Desenvolvimento de Sistemas, Versão 2.3.0 - Brasília: INEP, 2012. (MGDS-INEP, 2013).

Brasil. Instituto do Patrimônio Histórico e Artístico Nacional. MIDAS - Metodologia Iphan de Gestão de Demandas de Desenvolvimento Ágil de *Softwares*, Versão 1.0 - Brasília: IPHAN, 2013. (MIDAS-IPHAN, 2013).

Brasil. Tribunal de Contas da União. Acórdão no 2314/2013. Colegiado: Plenário. Relator: José Múcio Monteiro. Processo TC 010.663/2013-4. Ata 33/2013. Sumário: Levantamento de Auditoria. Conhecimento Acerca da Utilização de Métodos Ágeis nas Contratações para Desenvolvimento de *Software* pela Administração Pública Federal. Brasília, DF, Dou na Ata 33 - Plenário, de 28/08/2013. (Acórdão TCU 2314, 2013).

Brasil. Lei nº 8.666, de 21 de junho de 1993. Regulamenta o art. 37, inciso XXI, da Constituição Federal, institui normas para licitações e contratos da Administração Pública e dá outras providências. Diário Oficial da União, Brasília, DF, 22 jun. 1993. Disponível em: <http://www.planalto.gov.br/ccivil_03/Leis/L8666cons.htm>. (Lei 8.666, 1993)

Brasil. Lei Nº 10.520, de 17 de junho de 2002. Institui, no âmbito da União, Estados, Distrito Federal e Municípios, nos termos do art. 37, inciso XXI, da Constituição Federal, modalidade de licitação denominada pregão, para aquisição de bens e serviços comuns, e dá outras providências. (Lei 10.520, 2002).

Chiavenato, Idalberto. *Gestão de Pessoas:* e o novo papel dos recursos humanos nas organizações. Rio de Janeiro: Elsevier, 2004 – 3ª Reimpressão. (Chiavenato, 2004).

Cruz, Claudio Silva; Andrade, Edméia Leonor Pereira; Figueiredo, Rejane Maria da Costa. Processo de Contratação de Serviços de Tecnologia da Informação para Organizações Públicas. Ministério da Ciência e Tecnologia, PBQP *Software*, Brasília, 2011. (Cruz et al, 2011).

Crispim, Lisa; Gregory, Janet. Agile Testing: A Practical Guide for Testers and Agile Teams. Upper Saddle River, NJ: 2009. (Agile Testing, 2009)

De Castro, Eduardo José Ribeiro & Calazans, Angélica Toffano Seidel & Paldês, Roberto Ávila &. Guimarães, Fernando de Albuquerque. Engenharia de Requisitos: um enfoque prático na construção de *software* orientado ao negócio. Bokess, Florianópolis, SC, 2014. (De Castro et al, 2014).

Disciplined Agile Delivery. An agile process decision framework for the enterprise. Disponível em http://disciplinedagiledelivery.com/. Acessado em 08 de janeiro de 2015. (Disciplined Agile Delivery, 2014).

Druckman, Angela. How to Hold an Effective Backlog Grooming Session. Scrum Alliance, 2011. (Druckman, 2011).

Franco, E. F.: Um modelo de gerenciamento de projetos baseado nas metodologias ágeis de desenvolvimento de *software* e nos princípios da produção enxuta. Escola Politécnica da Universidade de São Paulo (Dissertação de Mestrado). 2007. (Franco, 2007).

Fadel, Aline Cristine & Silveira, Henrique da Mota. Metodologias ágeis no contexto de desenvolvimento de *software*: XP, Scrum e Lean. UNICAMP – Universidade Estadual de Campinas.FT – Faculdade de Tecnologia. 2010. (Fadel e Silveira, 2010).

Impact Mapping web site. Disponível em: <http://www.impactmapping.org/> Acesso em: maio de 2015. (Impact Mapping, 2015).

International Institute. Business Analysis.Guia BABOK: Um guia para o corpo de conhecimento de Análise de Negócios Versão 2.0. Toronto: International Institute of Business Analysis, 2011. (BABOK, 2011).

Instituto de Gerenciamento de Projetos (PMI). Um Guia do Conjunto de Conhecimentos em Gerenciamento de Projetos: Guia do PMBOK , 4a. edição, 2004, PMI. (PMBOK, 2004).

Institute Of Electrical and Eletronics Engineers (IEEE). Guide to the *Software* Engineering Body of Knowledge (SWEBOK). 2004 Version. (SWEBOK, 2004).

Institute Of Electrical and Eletronics Engineers (IEEE). Standard Glossary of *Software* Engineering Terminology; IEEE Std 610.12-1990, 1990. (IEEE610, 1990).

Matta, Villela da; Victoria, Flora. Personal & Professional Coaching: livro de metodologia. São Paulo: SBCoaching Editora, 2014. (Matta & Victoria, 2014).

Rook, Paul, E. Controlling *software* projects, IEEE *Software* Engineering Journal, 1986, pp. 7-16. (Rook, 1986).

Scaled Agile Framework web site. A Proven, Publicly Available Framework for Applying Lean / Agile Pratices at Enterprise Scale. Disponível em: <http://scaledagileframework.com/>. Acesso em out, 2014. (Scaled Agile Framework, 2014).

Schwaber, Ken; Sutherland, Jeff. The Definitive Guide to Scrum: The Rules of the Game. Scrum.org, 2013. Disponível em: <http://www.scrum.org/scrumguides/>. (Guia do *Scrum*, 2013).

Simões, Roberto. Scrumex - Expandindo a Aplicação e os Benefícios do *Scrum*. Disponível em: <http://scrumex.com.br>. Acesso em out, 2014. (Scrumex, 2014).

Tolfo, C & Forcellini, F. A & Vicentini, L. C. O Papel do Mentor e do Coach em Abordagens Ágeis: Descrição de um Caso. XXX Encontro nacional de Engenharia de produção. São Carlos, SP, Brasil, 12 a15 de outubro de 2010. (Tolfo et al, 2010).

ANEXO I - CONCEITOS E FUNDAMENTOS

Conceitos Gerais:

- **Adaptação**: Significa que toda vez que uma variação prejudicial ao projeto é identificada, o processo deve ser ajustado imediatamente, como forma de evitar outros desvios;
- **Brainstorming:** técnica dedicada a produzir um conjunto diverso de opções para um problema ou oportunidade;
- **DevOps:** é relação de interdependência entre os desenvolvedores de *software* e os profissionais de tecnologia da informação que viabilizam a operação do *software* e necessidades afins por meio de um sistema de produção e integrado de pessoas, informação, ativos de hardware, serviços, ferramentas e aplicativos, que apoiam as atividades de gestão de projetos, produção e gerência de requisitos de *software*, construção de *software*, prototipação de interfaces gráficas de usuário, construção e disponibilização do banco de dados, testes automatizados, inspeção automatizada da qualidade dos produtos, integração contínua dos componentes de *software*, disponibilização automática do *release* em ambiente apropriado, gestão informatizada dos ambientes de disponibilização, gestão de configuração de artefatos de engenharia e gestão de mudanças. Todas estas atividades de engenharia de *software* são desenvolvidas por meio de processos, métodos e ferramentas, alguns destes são informatizados, promovendo o princípio da impessoalidade e a diminuição do esforço da execução pela automação de processos operacionais. Os profissionais especializados em *DevOps* são responsáveis pela sua instalação, configuração, disponibilização, operação, conectividade, continuidade, segurança e manutenção. O *DevOps* exige comunicação estreita entre os profissionais, colaboração contínua e cooperação máxima (Scaled Agile Framework, 2014), (SWEBOK, 2004);
  - Veja mais informações em:
    - http://guide.agilealliance.org/subway.html
    - http://www.scaledagileframework.com/devops/
- **Cobertura de Testes**: os testes podem ser qualificados quanto a sua extensão e escopo. A cobertura é o limite de incidência do teste, podendo atingir os seguintes níveis de abstração: negócio (teste de aceitação), requisitos (teste de sistema), projeto (teste de integração) e código (testes unitários);
- **Complexidade Ciclomática:** Desenvolvida por Thomas J. McCabe em 1976, mede a quantidade de caminhos de execução independentes. Essa complexidade é computada através do grafo de fluxo de controle do programa: os nós do grafo correspondem a grupos indivisíveis de comandos, e uma aresta direcionada conecta dois nós se o segundo comando pode ser executado imediatamente após o primeiro. A complexidade ciclomática também pode ser aplicada a funções, módulos, métodos ou classes individuais de um programa. Uma estratégia de teste de verificação é testar cada caminho independente do programa, de modo que a quantidade de casos de teste será a complexidade ciclomática do código;
- **Débito técnico:** pendências técnicas introduzidas no código em virtude de o mesmo ter sido implementado sem o design ou cuidados necessários. Estas pendências podem estar relacionadas com: qualidade, bom design que permita fácil adaptação e manutenção, testes,,etc. A existência de débito técnico indica que a tarefa não foi totalmente concluída e que a equipe deverá ter mais trabalho futuramente em virtude destas pendências deixadas no projeto.
  - Veja mais informações em: http://martinfowler.com/bliki/TechnicalDebt.html http://martinfowler.com/bliki/TechnicalDebtQuadrant.html http://www.ontechnicaldebt.com/
- **Defeito:** problema no *software* que faz com que ele se comporte de forma distinta da esperada/desejada;
- **Desvio ou Impedimento:** é todo problema que afeta o ritmo de trabalho da equipe do projeto, podendo comprometer o alcance da meta estabelecida para a Iteração ou *Release*;
- **Definição de Pronto:** é um entendimento compartilhado entre os envolvidos no projeto (Instituição e contratada) do que significa trabalho completo, assegurando a transparência. O “Pronto” será utilizado para definir um nível de maturidade

e ou qualidade de entregáveis em cada grupo de atividades do processo. Por exemplo: artefatos de planejamento, requisitos, incremento de *software* (iteração) e versão funcional do *software* (*release)*;

- **Elevator Statement:** O seu principal objetivo é a facilitação do entendimento do projeto. Cauteriza o entendimento sobre as razões e aspectos relevantes do projeto na mente coletiva. A técnica pode ser executada em uma sala com toda a equipe do projeto, isolada das demais pessoas e do ambiente de trabalho. Em seguida, o facilitador faz uma explanação das necessidades, negócio e problema para a equipe do projeto, a fim de estabelecer uma visão clara da solução e familiarizar a equipe com o negócio. Caso seja necessário envolva algum *stakeholder* da área de negócio.

  Elementos mais relevantes de um projeto:

  - PARA [Área de Negócio]
  - QUE [Necessidade ou Oportunidade]
  - O [Nome da Solução]
  - É UM [Categoria da Solução]
  - QUE [Principal benefício]
  - DIFERENTEMENTE DO [Solução ou condição atual]
  - NOSSA SOLUÇÃO [Principal diferenciação]

Por fim, a construção de uma visão do projeto e a internalização de seu propósito e de seus aspectos mais relevantes são inciativas que contribuem com a clareza e a determinação;

Veja mais Informações: http://www.insistimento.com.br/empreendedorismo/auto-ajuda/como-criar-um-elevator-pitch-eficiente/

- **Épico:** é uma necessidade de negócio não refinada, correspondendo a algum caso de negócio. O épico do produto ou do produto-parte de uma solução informatizada (de um determinado problema) será decomposto em características macrofuncionais que o refinam. A solução deve agregar valor perceptível ao negócio do cliente. (*Scaled Agile Framework*, 2014);
- **Funcionalidade:** é a menor parte e mais elementar de um sistema informatizado, que agrega algum valor ao usuário. É uma história de usuário conforme definido pelo *framework* XP e que, para o Processo Unificado, seria equivalente à decomposição de um caso de uso ou mesmo à descrição do cenário de um caso de uso;
- **Grooming:** é uma técnica para refinamento e aprimoramento do *backlog* do produto, precisa de atenção, cuidados contínuos e ser organizado cuidadosamente (Druckman, 2011). Refinar o *backlog* abrange os seguintes procedimentos:
  1. A descoberta de novos itens, assim como alteração e remoção de itens antigos;
  2. Quebrar histórias muito grandes;
  3. A priorização dos itens do *backlog* (trazendo os mais importantes para o topo);
  4. Preparar e refinar os itens mais importantes para a próxima reunião de planejamento;
  5. Estimar e corrigir estimativas dos itens do *backlog* (*em caso de novas descobertas*);
  6. Incluir critérios de aceitação.

  A técnica contribui com a manutenção do foco, evitando distrações e desvios daquelas funcionalidades necessárias para o momento.

  - Veja Mais Informações:
    - http://guide.agilealliance.org/guide/backlog-grooming.html
- **Guias de Apoio:** são metodologias e especificações técnicas que auxiliam no processo de desenvolvimento e manutenção de sistemas de informação da instituição. São eles:
  - **Guia de Arquitetura:** define a estrutura ou organização dos componentes do programa (módulos), a maneira com a qual estes componentes interagem e a estrutura da informação que é usada por estes componentes na instituição;

- **Guia de Banco de Dados:** guia que concentra informações necessárias à criação e utilização de modelos de dados na instituição, tais como: modelo de dados corporativo definido, migração de dados, normalização de dados, sistema gerenciador de banco de dados e administração de dados;
- **Guia ou Roteiro de Métricas:** apresenta um roteiro de métricas, com base em regras de contagem, para vários tipos de projetos de desenvolvimento e de manutenção de sistemas de informação, promovendo o uso de métricas objetivas nos contratos de prestação de serviços desses projetos;
- **Guia de Segurança da Informação para Desenvolvimento de *Software*:** documento que apresenta requisitos de segurança para implementação e observação no desenvolvimento de sistemas de informação no âmbito da instituição, trata questões como: controles criptográficos e segurança de arquivos do *software*;
- **Guia de Configuração e Mudança:** documento que descreve o conjunto de atividades de controle de mudanças pela identificação dos produtos do trabalho que serão alterados, estabelecendo um relacionamento entre eles, definindo o mecanismo para o gerenciamento de diferentes versões destes produtos, controlando as mudanças impostas, e auditando e relatando as mudanças realizadas;
- **Guia de Interface do Usuário:** documento que define diretrizes da interface visual dos sistemas de informação, tais como: requisitos de usabilidade, navegabilidade, funcionalidades de tela e acessibilidade;
- **Guia de Testes de Sistemas:** documento que define diretrizes para realização do plano de testes. Define os tipos de testes, artefatos e perfis envolvidos nos testes;
- **Guia de Especificação de Requisitos:** documento que auxilia na definição de requisitos de um produto, sistematizando e normalizando o processo de especificação adotado pela instituição;

- **História de usuário (user story):** descrição curta de uma característica do produto contada na perspectiva do usuário, utilizando uma linguagem comum ao negócio.
  - Veja mais informações em:
    - http://www.extremeprogramming.org/rules/userstories.html http://www.mountaingoatsoftware.com/topics/user-stories http://guide.agilealliance.org/guide/user-stories.html
- **Impact Mapping:** técnica que auxilia a equipe a entregar *software* focado em objetivos de negócio, nas partes interessadas e suas necessidades.
  - Veja mais informações:
    http://www.impactmapping.org/
    http://gojko.net/papers/effect_maps.pdf
- **Incremento de *Software*:** acréscimo de funcionalidades do produto de *software* final que é gerada a cada iteração. O incremento de *software* deve ser algo observável e, preferencialmente funcional, para que o dono do produto e demais usuários possam experimentá-lo da forma mais realista possível;
- **Inspeção:** Todos os aspectos do processo de entrega que possam impactar o resultado final do projeto devem ser inspecionados frequentemente, para que qualquer variação prejudicial possa ser identificada e corrigida o mais rápido possível;
- **Iteração:** um ciclo de desenvolvimento, de duração fixa e curta, que corresponde a uma subdivisão da *release* e que produz uma versão estável e executável do produto de *software*;
  - Veja mais informações em:
    http://guide.agilealliance.org/guide/iteration.html
- **Meta:** breve descrição em função dos objetivos de negócio e prazo que a equipe pretende alcançar durante um *release* ou iteração;
- **Produto:** resultado final de um projeto, cuja finalidade é responder a demandas associadas ao *software*, ajustes na or-

ganização, necessidade de um ou mais clientes, avanços tecnológicos ou obrigatoriedades legais. O termo produto está relacionado a um produto físico, serviço ou associação de ambos. Independente da origem da demanda, um produto sempre é destinado a um cliente ou grupo de clientes, internos ou externos (Scrumex, 2014).

- **Product Vision Box:** Nesta técnica, o facilitador captura informações sobre o problema ou necessidades para delinear a solução. Os *stakeholders* e o dono do produto são responsáveis pela transmissão do conhecimento das necessidades e do negócio. A técnica permite elencar as necessidades, aspectos do negócio, características e propósitos, e, assim, tecer os fundamentos e a anatomia da solução.

  Os interessados pela solução ajudarão a construir a visão do produto, esclarecendo não somente uma forma de resolver o problema proposto, como também elucidando desafios, dificuldades, riscos e expectativas.

  São elementos importantes do *Product Vision Box*, mas não se limitam a:
  - Visão da Solução;
  - Proposta de valor e o alinhamento estratégico gerencial;
  - Modelo de negócio;
  - Características do produto;
  - Exposição dos problemas e necessidades;
  - Oportunidades exploradas.

  Finalmente, é possível, com esta técnica, extrair a visão que os donos do produto têm, reforçando o *briefing* (dossiê), o que garante, ao final de uma entrega, um produto funcional.

  - Veja mais informações em: http://www.techrepublic.com/blog/tech-manager/define-the-vision-for-your-it-project-with-this-exercise/710

- **Release:** conjunto de funcionalidades extraídas do *backlog* do produto. Representa entrega de um *software* funcional incluindo funcionalidades de uma ou mais Iterações;
- **Requisitos Funcionais:** é a descrição do comportamento e a informação que a solução gerenciará. Descrevem capacidades que o sistema será capaz de executar em termos de comportamentos e operações – ações ou respostas específicas de aplicativos de tecnologia da informação (BABOK, 2011), (IEEE610, 1990);
- **Requisitos não Funcionais:** são condições que não se relacionam diretamente ao comportamento ou funcionalidade da solução, mas descrevem condições ambientais sob as quais a solução deve permanecer efetiva, ou qualidades que os *softwares* precisam possuir. Também são conhecidos como requisitos de qualidade ou suplementares. Podem incluir requisitos relacionados à capacidade, velocidade, segurança, disponibilidade, arquitetura da informação e apresentação da interface com o usuário (BABOK, 2011);
- **Regras Transacionais ou de Execução:** são determinadas condições de comportamento ou estado que o *software* deve assumir, também são conhecidas como regras de negócio. Os dados manipulados pelo *software* poderão assumir algum estado em determinadas condições definidas em uma regra (De Castro et al, 2014);
- **Requisitos de Dados:** são estruturas de dados abstratas que definem qualidades ou características para as entidades do negócio. É qualquer tipo de dados necessário e manipulável pelo *software* (De Castro et al, 2014);
- **Refinamento dos Requisitos:** considerando os aspectos do desenvolvimento ágil, define-se refinamento dos requisitos (retrabalho) como sendo os refinamentos referentes a funcionalidades que estão em desenvolvimento dentro de uma mesma *release*, ou seja, mudanças entre as iterações da *release*. Anexo V apresenta um quadro sobre itens de tipos de alterações em funcionalidades em métodos ágeis;
- ***Scrum:*** é um framework de desenvolvimento ágil de *software* iterativo e incremental para o gerenciamento de desenvolvimento de produtos (Guia do *Scrum*, 2003);
  - Veja mais informações em:

http://guide.agilealliance.org/subway.html

- ***Stakeholder*:** indivíduo ou organização que tem um direito, ação, declaração ou interesse no *software* em desenvolvimento; envolvido; interessado (PMBOK, 2004);
- ***Time-box*:** a tradução literal é "caixa de tempo" e deve ser entendido como um intervalo de tempo para realização de uma atividade. É uma técnica utilizada para assegurar que atividades importantes do projeto sejam cumpridas e com o menor desperdício possível;
  - Veja mais informações em:
    http://guide.agilealliance.org/guide/timebox.html
- **Teste de *Software*:** consiste numa verificação dinâmica do comportamento de um programa em um conjunto finito de casos de teste contra o comportamento esperado.(SWEBOK, 2004), (IEEE610, 1990);
- **Teste de Operação:** uma versão do *software* é disponibilizada no ambiente de produção para que um grupo restrito de usuários utilizem em circunstâncias reais. Aspectos funcionais e comportamentais do *software* são avaliados em uma implantação piloto no ambiente de produção (IEEE610, 1990);
- **Teste de Aceitação:** teste escrito sob a perspectiva do cliente [BECK04]. Esse tipo de teste, geralmente, especifica o resultado esperado após a execução de determinada funcionalidade (IEEE610, 1990);
- **Teste de Sistema**: o objetivo é executar as funcionalidades do sistema sob a ótica de quem utilizará, explorando as funcionalidades em busca de inconformidades em relação aos objetivos e especificações funcionais. Os testes são executados em condições operacionais similares, por exemplo, ambiente, interfaces sistêmicas e massas de dados, àquelas que um usuário utilizará na sua rotina de operação do *software* (IEEE610, 1990);
- **Teste de Integração:** encontrar falhas provenientes da integração interna dos componentes de um *software*, ele conduz ao descobrimento de possíveis falhas, erros e defeitos associados à interface do *software*. A critério da gerência de projetos, os testes de comunicação entre interfaces com outros *software*s (integração entre eles) poderá ser realizado (IEEE610, 1990);
- **Teste de Unidade:** teste escrito sob a perspectiva do programador [BECK04], aplicado sobre a menor unidade de execução do código. Cada caso de teste deve avaliar um possível comportamento do código (IEEE610, 1990);
- **Teste de Desempenho:** teste realizado para avaliar a adequação de um sistema ou componente com determinados requisitos de desempenho (IEEE610, 1990);

PAPÉIS DA EQUIPE DA INSTITUIÇÃO:

- **Analista de Infraestrutura de TI:** é o representante da área de infraestrutura para o projeto;
  - **Responsabilidades:**
    - Definir e executar projetos de implantação de infraestrutura de TI;
    - Identificar e instalar infraestrutura de hardware, *software* e serviços para ambientes de TI (Teste, Homologação, Produção);
    - Planejar, executar, acompanhar e controlar os projetos de infraestrutura de TI, identificando riscos e impactos e garantindo o cumprimento das tarefas e prazos;
    - Desenvolver e analisar indicadores de desempenho a fim de mensurar os objetivos e desempenho da área;
    - Garantir a padronização dos processos internos, disponibilizando internamente os documentos relativos à área de infraestrutura.
- **Analista de Qualidade:** representa diferentes perfis ligados a avaliação de qualidade de *software*, tais como: Analista de

Teste, Arquiteto de *Software*, Analista de Banco de Dados, Analista de Configuração e Mudança e outros.

- **Responsabilidades:**
  - Avaliar e definir processos de verificação da qualidade dos produtos entregues pela Contratada (artefatos, incremento de *software* e *release*);
  - Realizar testes de sistemas.

- **Analista de Métricas:** é o responsável pelos serviços de medição de *software*.
  - **Responsabilidades:**
    - Realizar medições de *software* utilizando a métrica definida no contrato com a empresa de desenvolvimento de *software*.
- **Cliente:** representante da área de negócio.
  - **Responsabilidades:**
    - É responsável pela solicitação e homologação dos serviços de desenvolvimento de *software*.
- ***Coach*:** profissional experiente nas práticas do processo ágil que zela pela sua correta execução. Deve ter profundo conhecimento das práticas ágeis adotadas, pois é ele que guiará os outros envolvidos no projeto a executar o processo e práticas de forma adequada. Esse papel pode ser exercido pelo líder de projeto.
  - **Responsabilidades:**
    - Conhecer e disseminar processo e práticas de desenvolvimento ágil na instituição;
    - Orientar na execução correta das práticas ágeis;
    - Comunicar e negociar;
    - Trabalhar em equipe.
- **Dono do Produto (*Product Owner*):** representante da área de negócio com conhecimento suficiente para definir e priorizar requisitos do negócio e responder aos questionamentos da equipe de desenvolvimento. É o representante do cliente e sua atuação tem por finalidade garantir a entrega do valor esperado através do produto final do projeto.
  - **Responsabilidades:**
    - Estabelecer a visão do produto;
    - Aprovar o Roadmap de releases do produto;
    - Gerenciar o backlog do produto, fornecendo e priorizando os requisitos;
    - Definir prioridades de negócio;
    - Conhecer e transmitir o processo de negócio e seus objetivos;
    - Detalhar os requisitos de cada funcionalidade;
    - Aceitar ou rejeitar produtos e serviços;
    - Gerenciar as expectativas dos stakeholders;
    - Resolver os itens de backlog não planejados;
    - Aprovar mudanças no projeto;
    - Preparar e Realizar Treinamentos;
    - Dar apoio à equipe da contratada de desenvolvimento durante todo o projeto;
    - Obs: O dono do produto é uma pessoa e não um comitê. O dono do produto pode representar o desejo de um comitê no *backlog* do produto, mas aqueles que quiserem uma alteração nas prioridades dos itens de *backlog* devem convencer o dono do produto.
- **Líder de Projeto:** é responsável do órgão ou entidade pelo projeto, que se relaciona com todos os envolvidos e atividades do projeto.
  - **Responsabilidades:**
    - Orientar e acompanhar a execução das atividades do projeto;

- Recebe, comunica e resolve desvios e impedimentos do projeto;
- Comunicar o andamento do projeto aos interessados;
- Acompanhar o cronograma das atividades do planejamento, iterações e *releases* do projeto e das ordens de serviço;
- Auxiliar na definição e acompanhamento dos riscos do projeto;
- Avaliar junto à área de TI as necessidades de recursos;
- Avaliar o andamento do(s) projeto(s);
- Gerenciar treinamentos da equipe de suporte e sustentação;
- Atualizar o portfólio de projetos, com o devido status do(s) projeto(s);
- Gerar informações de desempenho do projeto;
- **Obs:** Ressalta-se que na estrutura de pessoal das organizações públicas existe o cargo em comissão com a denominação Gerente de Projeto. Este cargo pertence ao grupo de Direção e Assessoramento Superiores (DAS). Por exemplo, conforme Decreto no 7.063/2010, existem na estrutura regimental do Ministério do Planejamento, Orçamento e Gestão os cargos DAS 101.4 – Gerente de Projetos e DAS 101.5 – Diretor de Programa. Para que não ocorra conflito com as nomenclaturas existentes, utilizaremos aqui a denominação Líder de Projetos.

- **Usuário:** tipo de *stakeholder* que usa o produto de *software*. Pode ser interno ou externo a instituição;
  - **Responsabilidades:**
    - Auxiliar na definição dos itens de backlog;
    - Auxiliar na homologação e validação das entregas do projeto;

PAPEIS RELACIONADOS À INSTRUÇÃO NORMATIVA 04/2014:

- Gestor do Contrato: verificar (IN04, 2014).
- Fiscal Requisitante do Contrato: verificar (IN04, 2014).
- Fiscal Técnico do Contrato: verificar (IN04, 2014).
- Fiscal Administrativo do Contrato: verificar (IN04, 2014).

PAPÉIS DA EQUIPE DA CONTRATADA DE DESENVOLVIMENTO DE *SOFTWARE*

- **Preposto:** verificar (IN04, 2014).
- **Mestre *Scrum* (Scrum Master):** é um dos representantes da contratada de desenvolvimento de *software*. Deve trabalhar para o sucesso do projeto através da entrega de valor para o cliente, viabilizada pela atuação eficaz da equipe de desenvolvimento e apoio ao dono do produto.
  - **Responsabilidades:**
    - Remover impedimentos da equipe de desenvolvimento (problemas técnicos, administração de conflitos, itens não planejados);
    - Auxiliar o dono do produto com técnicas para o gerenciamento efetivo do *backlog* do produto;
    - Comunicar claramente a visão, objetivo e itens do *backlog* do produto para a equipe de desenvolvimento;
    - Ser o facilitador da equipe de desenvolvimento, garantindo sua plena produtividade;
    - Garantir a colaboração entre os diversos papéis e funções da equipe de desenvolvimento;
    - Facilitar eventos e reuniões do *Scrum* conforme exigidos ou necessários;

    - Aplicar e disseminar valores e práticas *Scrum*.

- **Equipe de Desenvolvimento (Time de Desenvolvimento):** é um grupo de cinco a nove integrantes da contratada de desenvolvimento de *software*, que deve possuir uma característica multifuncional, auto-organizados, contando com analistas, arquitetos, programadores, testadores, administrador de banco de dados entre outros. Individualmente os integrantes da Equipe de desenvolvimento podem ter habilidades especializadas e área de especialização, mas a responsabilidade pertence a toda equipe de desenvolvimento.
    - **Responsabilidades:**
        - Fazer as estimativas necessárias;
        - Definir as tarefas que serão realizadas;
        - Desenvolver o produto;
        - Realizar repasses de conhecimento para as equipes de suporte e sustentação;
        - Garantir a qualidade do produto;
        - Apresentar o produto ao cliente.

- **Gerente de Relacionamento (opcional):** realiza o acompanhamento de todo o ciclo de atendimento da ordem de serviço garantindo as entregas conforme compromissos firmados.
    - **Responsabilidades:**
        - Acompanhar o atendimento da Ordem Serviço no âmbito da contratada;
        - Identificar desvios e providenciar as ações necessárias para correção de rumos dentre todos os envolvidos;
        - A partir das necessidades do Dono do Produto, auxiliar no mapeamento de soluções;
        - Acompanhar o cronograma geral de atendimento e os compromissos firmados entre os envolvidos.

AGRUPAMENTOS EM EQUIPES:

- **Equipe do Projeto:** todos os envolvidos no projeto, equipe da instituição e equipe da contratada.

PRINCIPAIS ARTEFATOS:

- ***Backlog* da Iteração:** é um subconjunto do *backlog* do produto, priorizado na reunião de planejamento pelos clientes e dono do produto e que representa os itens que serão desenvolvidos em uma determinado Iteração;
- ***Backlog* do Produto:** é uma lista ordenada de tudo que deve ser necessário no produto, e é uma origem única dos requisitos para qualquer mudança a ser feita no produto. O *backlog* do produto lista todas as características, funções, requisitos, melhorias e correções que formam as mudanças que devem ser feitas no produto;
- **Documento de Visão:** documento que contém as informações identificadas durante o entendimento da demanda, tais como objetivos de negócio, características-chaves, aspectos tecnológicos e riscos, utilizados para orientar o desenvolvimento do produto de *software*;
- **História de Usuário:** descrição curta de uma característica-chave do produto contada na perspectiva do usuário, utilizando uma linguagem comum ao negócio e testes, que podem ser utilizados para determinar quando a história estará completa;
- **Gráfico *Burndown*:** gráfico que mostra o esforço restante para a conclusão da iteração, bem como mostrar o quão próximo ou distante a equipe de desenvolvimento está de atingir a meta;
- **Plano do *Release*:** descreve informações importantes do *release*, tais como: metas do *release*, quantidade e duração de Iterações, equipe do projeto, data estimada da entrega, valor estimado do *release*, premissas, riscos e impedimentos;

- **Plano da Iteração:** descreve informações importantes da Iteração, tais como: Meta da Iteração, data inicial e final da Iteração, definições de pronto, itens de *backlog* selecionados e outros;
- **Quadro de Tarefas:** tem por objetivo transformar o trabalho em andamento da iteração visível em um quadro para toda equipe, criando um sinal visual que indica que uma nova tarefa pode ou não ser iniciada, se iniciada mostra seu andamento e se o limite acordado de tarefas, para cada iteração, está sendo respeitado.
- ***Roadmap*:** é uma visão global das necessidades do produto e uma ferramenta valiosa para o planejamento e organização da jornada de desenvolvimento de produtos. O dono do produto cria o *roadmap* de produtos, com a ajuda da equipe de desenvolvimento. O roteiro é usado para categorizar os requisitos, para priorizá-los, e para determinar um calendário para a sua liberação.

REUNIÕES DA ITERAÇÃO:

- **Reunião de Planejamento da Iteração:** consiste em uma reunião realizada no início de cada Iteração para planejar e definir o que será entregue;
- **Reunião de Demonstração da Iteração:** consiste em uma reunião executada no final da Iteração para demonstrar o incremento de *software* produzido;
- **Reunião de Retrospectiva da Iteração:** consiste em uma reunião realizada no fim de cada Iteração, para registrar as lições aprendidas e fazer os ajustes possíveis para a próxima iteração, proporcionando assim, a melhoria contínua do processo.

ANEXO II - EXEMPLIFICAÇÃO DE ITENS DA CADEIA DE VALOR DO PRODUTO

Este anexo apresenta exemplos de itens da cadeia de valor do produto apresentado no modelo de referência da figura 2. O fluxo dos itens da cadeia de valor mais importantes sãomostrados na figura 14, são eles: objetivos de negócio, características-chaves do produto, histórias de usuário, tarefas, incremento de *software* e *release*.

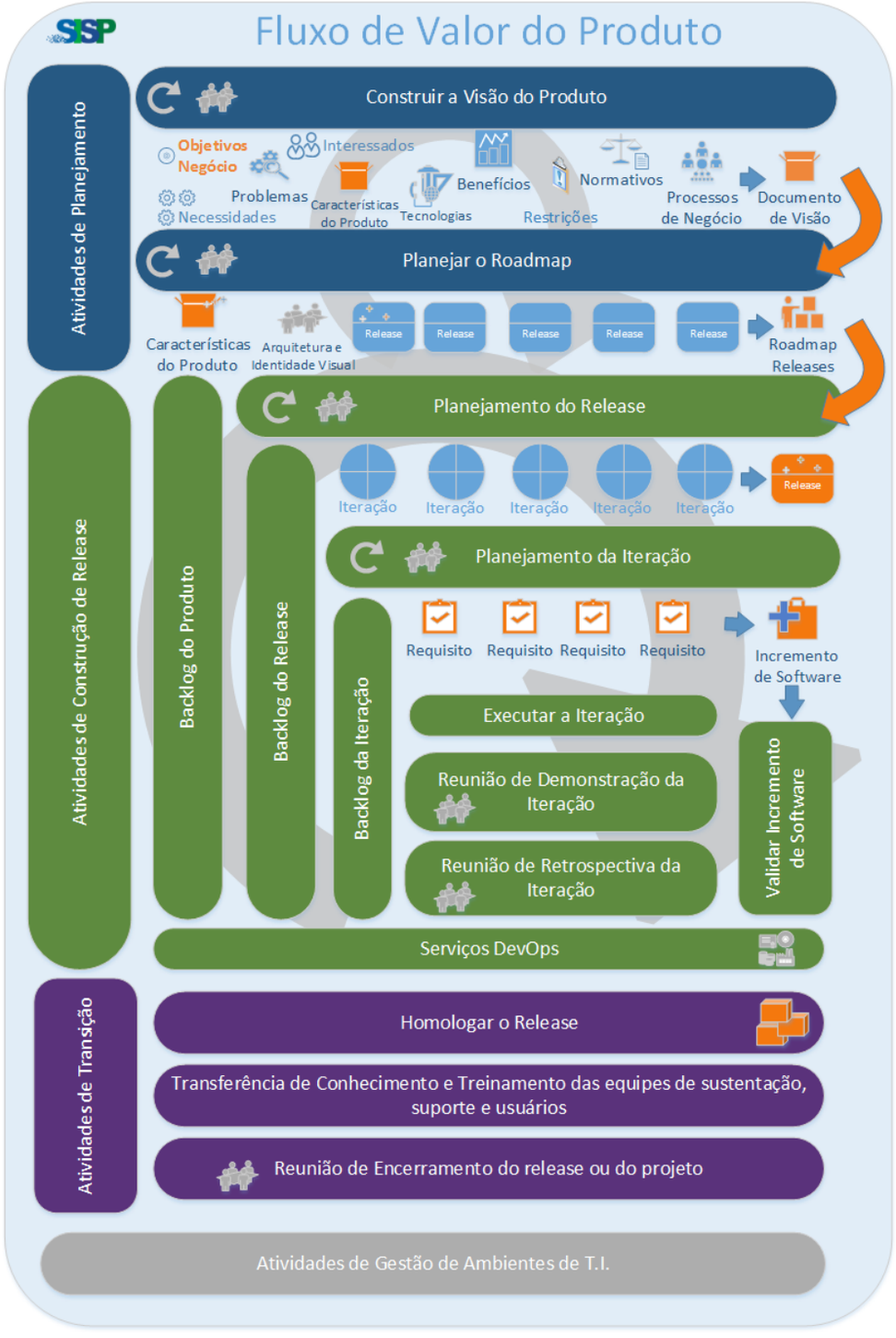

Figura 14. Fluxo de Valor do Produto (Baseado em Scrumex, 2014).

Contexto do Problema: Dificuldades de gerir contratos da administração pública.

Obs: este contexto do problema foi criado apenas para exemplificar os itens da cadeia de valor.

Visão do Produto: entende-se que existe um **problema** a ser resolvido que é a dificuldade em controlar pagamentos efetuados por serviços prestados para a administração pública. Segue no quadro abaixo a lista de exemplos de itens da cadeia de valor.

Item de Valor	Exemplo
Problema	Não existe controle dos pagamentos efetuados de cada contrato.
Objetivo Específico de Negócio	Gestão eficiente de todos os pagamentos dos contratos.
Características-chaves do Produto	Controlar pagamentos efetuados no mês e no ano por contrato.
História de Usuário	História de usuário: Cadastrar notas fiscais relativas aos serviços prestados por contratada. Regra de criação de história de usuário: Como / Eu gostaria / para (especificando requisito) O gestor do contrato / deseja ter um registro persistente de todas as notas fiscais de pagamento (número da nota, valor, fornecedor, data de emissão, data de validade, data da liquidação) para saber o total pago por período; Regra transacional (Critério de aceitação): toda vez que o valor da nota fiscal for superior a R$ 100.000,00 deve existir autorização do gestor do departamento para liberar o pagamento;
Tarefas	Tarefas para implementação da história de usuário: Criar *scripts* para criação do modelo de dados; Implementar as classes de cadastro de notas fiscais; Criar testes unitários; Executar testes unitários; Versionar artefatos.
Incremento de *Software*	*Software* funcional contendo histórias de usuário implementadas.

ANEXO III - AVALIAÇÃO DE RISCOS DO ACÓRDÃO DO TCU 2314/2013.

Este anexo apresenta uma relação entre os riscos levantados pelo acórdão e este guia. A matriz relaciona os riscos e as respectivas aplicações do guia.

Para fins didáticos, o TCU dividiu os riscos em três grupos distintos: processos, produtos e pessoas. Portanto, a avaliação dos riscos, em relação ao guia, também utilizará esse agrupamento. Para cada risco foram definidas as seguintes possibilidades de avaliação: mitigado, não mitigado ou não se aplica ao contexto deste guia.

Os riscos mitigados e não mitigados são aqueles existentes dentro do universo do projeto ou das atividades de gestão de ordem de serviço e acompanhamento de projetos. O risco mitigado é aquele tratado pelo modelo de referência com suas atividades. O risco não mitigado não foi tratado pelo modelo de referência com suas atividades. Para os riscos em que "Não se aplica" ocorre quando o risco transcende o universo dos grupos de atividades específicas do projeto, as atividades de gestão de ordem de serviço ou acompanhamento de projeto registrados no modelo de referência.

É importante destacar que entre os riscos elencados, segundo o próprio TCU, alguns não são inerentes ao uso de metodologias ágeis, podendo ocorrer também em metodologias tradicionais de desenvolvimento de *software* (Acórdão TCU 2314, 2013).

Riscos Relativos a Processos		
Risco	**Descrição**	**Avaliação do Risco em relação ao Guia**
Risco 1	contratação de desenvolvimento de *software* com adaptação de metodologia ágil que desvirtue sua essência.	Não Mitigado: Embora o guia permita o planejamento adaptativo, por meio do modelo de referência com as suas atividades, a aplicação dos métodos ágeis dependem de uma adaptação na Administração Pública para ser eficiente.
Risco 2	alteração da metodologia ágil adotada no instrumento convocatório no decorrer da execução contratual.	Não se Aplica: Não faz parte da missão deste guia tratar a transição entre modelos de processo de desenvolvimento ou paradigmas de engenharia de *software*, tanto para contratos existentes, como para novas contratações.
Risco 3	ausência de definição dos artefatos ou alteração dos artefatos exigidos da contratada no instrumento convocatório durante a execução contratual.	Não se Aplica: Não faz parte da missão deste guia tratar a transição entre modelos de processo de desenvolvimento ou paradigmas de engenharia de *software*, tanto para contratos existentes, como para novas contratações.
Risco 4	exigência de artefatos desnecessários ou que se tornam obsoletos rapidamente.	Mitigado: o guia possui um conjunto de artefatos suficientes e adequados para cobrir todas as etapas de construção do *software*, desde o planejamento até as atividades de transição (ambiente de produção).
Risco 5	utilização de contrato para desenvolvimento de *software* por metodologias tradicionais para desenvolvimento por métodos ágeis.	Não se Aplica: Não faz parte da missão deste guia, tratar a transição entre modelos de processo de desenvolvimento ou paradigmas de engenharia de *software*, tanto para contratos existentes, como para novas contratações.

Quadro 2. Riscos Relativos a Processos.

Riscos Relativos a Pessoas		
Risco	**Descrição**	**Avaliação do Risco em relação a esse Guia**
Risco 6	falta de comprometimento ou colaboração insatisfatória do responsável indicado pela área de negócios (*Product Owner*) no desenvolvimento do *software*.	Não Mitigado: as práticas ágeis adotadas no guia tem uma dependência do dono do produto (Product Owner). Mas indicamos medidas para minimizar o risco, como a realização de treinamentos, o apoio constante ao P.O. e a definição de artefatos para facilitar a participação do dono do produto (P.O.).
Risco 7	falta do conhecimento necessário do indicado pela área de negócios (*Product Owner*) para o desenvolvimento do *software*.	Não mitigado: Não é missão do guia propor técnicas para a seleção de *stakholders* ou dono do produto. Qualquer metodologia seria impactada pela falta de conhecimento dos envolvidos da área de negócios.
Risco 8	excessiva dependência da visão do indicado pela área de negócios (*Product Owner*).	Não Mitigado: entende-se que o dono do produto (P.O.) apenas centraliza a tomada de decisão. Mas sua visão é determinada pelo conjunto de clientes e usuários da solução. O dono do produto pode atuar como representante de um comitê de clientes, o qual delibera questões sobre o negócio em relação ao projeto.
Risco 9	equipe da empresa contratada não ter expertise em desenvolvimento de *software* com métodos ágeis.	Não se Aplica: o guia não apresenta critérios de seleção de fornecedor, mas entende que é um fator de sucesso para a entrega de projetos de *software*. O que ele traz é um modelo de construção de projetos de *software*, incluindo um modelo de execução de ordens de serviços com recomendações e critérios de avaliação dos serviços entregues pela contratada.
Risco 10	dificuldade de comunicação entre a equipe de desenvolvimento da contratada com o indicado pela área de negócios (*Product Owner*).	Não Mitigado: qualquer metodologia estaria impactada pela dificuldade de comunicação com o(s) indicado(s) pela área de negócios. Mas entende-se que este risco é aumentado com essa metodologia.

Quadro 3. Riscos Relativos a Pessoas.

Riscos Relativos a Produtos		
Risco	**Descrição**	**Avaliação do Risco em relação a esse Guia**
Risco 11	alteração constante da lista de funcionalidades do produto.	Não Mitigado: vários fatores podem implicar este risco, tais como: Falta de capacidade técnica e produtiva da contratada; Falta de conhecimento do dono do produto (P.O.); Falta de compromisso do dono do produto (P.O.) com o projeto; Alta volatilidade nos cenários e ambientes da instituição.
Risco 12	iniciação de novo ciclo sem que os produtos construídos na etapa anterior tenham sido validados.	Não Mitigado: entende-se que o modelo de execução de contratos ou modelo de execução interna do processo de desenvolvimento ágil é quem mitigará este risco.
Risco 13	falta de planejamento adequado do *software* a ser construído.	Mitigado: o guia apresenta um grupo de atividades no inicio e durante o projeto para garantir o planejamento e sua atualização durante o projeto.
Risco 14	pagamento pelas mesmas funcionalidades do *software* mais de uma vez, em virtude de funcionalidades impossíveis de serem implementadas em um único ciclo, ou em virtude da alteração de funcionalidades ao longo do desenvolvimento do *software*.	Não Mitigado: o guia não fornece um modelo que mitigue este risco.
Risco 15	não disponibilização do *software* em ambiente de produção para a utilização e avaliação dos reais usuários.	Mitigado: O guia define atividades e orientações para uma estratégia de disponibilização do *software* funcional em ambiente de produção. Cabe a Instituição Contratante determiná-la.
Risco 16	forma de pagamento não baseada em resultados.	Mitigado: o guia propõe um modelo de execução de ordens de serviço com recomendações para liquidação dos serviços prestados somente após a entrega e avaliação dos mesmos (incremento e *Release* de *Software*).

Quadro 4. Riscos Relativos a Produtos.

ANEXO IV - QUADRO DAS NÃO CONFORMIDADES.

Quadro das não conformidades			
Hipótese de Causa	**Descrição**	**Impacto no recebimento (IN04/2014 / Homologação)**	**Qual consequência em termos de métricas ?**
Especificação de *software* não atendida	Incrementos de *software* que não satisfaçam à especificação dos requisitos de *software* acordados entre contratante e contratada seja por omissão de implementação, seja por implementação inválida.	Impeditivo	Pendência
Erros e Defeitos de *software*	Qualquer inconformidade que impeça o correto funcionamento do *software* entregue	Impeditivo	Pendência
Débito técnico	Qualquer inconformidade que não impeça o funcionamento do *software*	Permissivo	-
Débito técnico reincidente	Qualquer promessa ou compromisso **reincidentemente não cumpridos** para resolver inconformidade que não impeça o funcionamento do *software*.	Impeditivo	Pendência

Quadro 5. Quadro das não conformidades

ANEXO V - QUADRO DOS TIPOS DE ALTERAÇÕES EM FUNCIONALIDADES EM MÉTODOS ÁGEIS.

Tipos de Alterações em Funcionalidades em Métodos Ágeis	Descrição	Impacto no recebimento (IN04/2014 / Homologação)	Qual consequência em termos de métricas ?
Refinamento	Resultante da evolução natural dos requisitos ou aprimoramento do entendimento da história de usuário (tela/funcionalidades), provocada pelo aprofundamento, detalhamento e complementação de requisitos durante o processo de desenvolvimento ágil.	Impeditivo/ Permissivo	Ver Roteiro de Métricas de *Software* do SISP.
Mudança de Escopo	É resultante da evolução das políticas, estratégias, legislação, objetivos específicos de negócio, necessidades ou processos de negócio do Órgão Público ou Entidade Pública.	Impeditivo/ Permissivo	Ver Roteiro de Métricas de *Software* do SISP.

Quadro 6. Quadro dos Tipos de Alterações em Funcionalidades em Métodos Ágeis.

Contribuições e Sugestões de Melhoria

Contribuições e Sugestões de Melhoria para o Guia de Projetos de Software com Práticas de Métodos Ágeis podem ser enviadas para:

- Central de Serviços do SISP: http://c3s.sisp.gov.br/cau/ ou
- E-mail: sisp@planejamento.gov.br

Secretaria de
Logística e Tecnologia da Informação

Ministério do
Planejamento