TRIBUNAL REGIONAL ELEITORAL DO PIAUI

Portaria Presidência Nº 258/2025 TRE/PRESI/DG/ASSDG, de 26 de maio de 2025

Altera o Processo de Software previsto na Portaria nº 839, de 14 de agosto de 2018, mediante instituição de novo Manual do Processo de Desenvolvimento de Software, e revoga a Portaria Presidência Nº 794/2022 TRE/PRESI/DG/ASSDG, de 25 de agosto de 2022.

O PRESIDENTE DO TRIBUNAL REGIONAL ELEITORAL DO PIAUÍ, no uso de suas atribuições legais e regimentais,

CONSIDERANDO a necessidade de atualização do processo de desenvolvimento de software no âmbito do Tribunal Regional Eleitoral do Piauí;

CONSIDERANDO o disposto na Resolução CNJ nº 370, de 28 de janeiro de 2021, que “Estabelece a Estratégia Nacional de Tecnologia da Informação e Comunicação do Poder Judiciário — ENTIC-JUD”;

RESOLVE:

Art. 1º Alterar o processo de desenvolvimento de software regulamentado pela Portaria Presidência Nº 839/2018 TRE/PRESI/DG/STI, de 14 de agosto de 2018, que passa a vigorar conforme o Manual do Processo de Desenvolvimento de Software, anexo único desta Portaria.

Art. 2º Fica revogada a Portaria Presidência Nº 794/2022 TRE/PRESI/DG/ASSDG, de 25 de agosto de 2022.

Art. 3º Esta Portaria entra em vigor na data de sua publicação.

Desembargador SEBASTIÃO RIBEIRO MARTINS

Presidente do TRE-PI

Documento assinado eletronicamente por **Sebastião Ribeiro Martins**, **Presidente**, em 27/05/2025, às 11:19, conforme art. 1º, § 2º, III, "b", da Lei 11.419/2006.

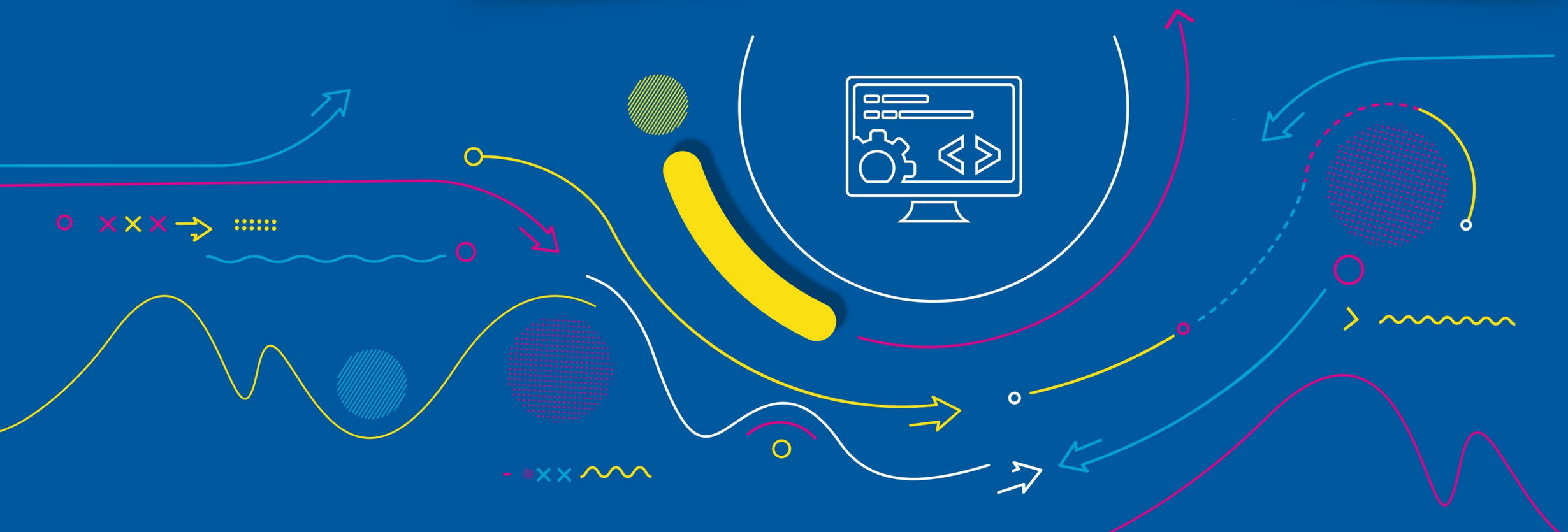
A autenticidade do documento pode ser conferida no site https://sei.tre-pi.jus.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 informando o código verificador **0002420021** e o código CRC **9CA59E91**.


2024
PRÊMIO
CNJ DE
QUALIDADE
Ouro

TRE-PI

STI
Secretaria da Tecnologia da Informação

MANUAL DO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE



TRIBUNAL REGIONAL ELEITORAL DO PIAUÍ
Praça Desembargador Edgar Nogueira s/nº, Centro Cívico
CEP 64000-830 Teresina – Piauí
Telefone: (86) 2107-9700
E-mail: sti@tre-pi.jus.br

Elaboração:
Paulo das Neves e Silva Júnior

Revisão e Alteração para versão 3.0:
CODIN
Titular: Rosemberg Maia Gomes

SEDESC – Seção de Desenvolvimento de Soluções Corporativas
Titular: Paulo das Neves e Silva júnior

Validação:
Comitê Gestor de Tecnologia da Informação

Aprovação:
Comitê Gestor de Tecnologia da Informação

Editoração eletrônica e capa:
Breno Ponte de Brito – SECOM

Ficha catalográfica:
Jovita Maria Gomes Oliveira – SEJUB

Disponível também em: <http://www.tre-pi.jus.br>

TRE-PI | STI Secretaria da Tecnologia da Informação

Dados Internacionais de Catalogação na Publicação
Tribunal Regional Eleitoral do Piauí - Biblioteca Des. Cristino Castelo Branco

Brasil. Tribunal Regional Eleitoral (PI).
Manual do processo de desenvolvimento de software [recurso eletrônico] / Tribunal Regional Eleitoral do Piauí. - Dados eletrônicos (52 páginas). - Teresina: Tribunal Regional Eleitoral do Piauí, 2025.

Elaboração: Seção de Desenvolvimento de Soluções Corporativas (SEDESC / CODIN / STI).
Revisão e alteração para versão 2.0: Coordenadoria de Desenvolvimento e Infraestrutura (CODIN / STI).
Versão eletrônica (PDF)
Modo de acesso: internet
<https://www.tre-pi.jus.br/>

1. Desenvolvimento de sistemas – Manual. I. Brasil. Tribunal Regional Eleitoral (PI). Secretaria de Tecnologia da Informação. II. Título.

CDD: 005.1

SUMÁRIO

Apresentação

Consistindo em uma das grandes forças que movem a administração do Tribunal Regional Eleitoral do Piauí, a Tecnologia da Informação é alvo frequente das demandas internas das unidades administrativas deste órgão, as quais, sempre em busca de excelência na prestação dos seus serviços, buscam as ferramentas de TI como suporte para a realização de suas atividades.

Especificamente no contexto de software, há recomendações do TCU, na área de Governança de TI, quanto à institucionalização de uma metodologia, a qual defina padrões a serem seguidos para o desenvolvimento de soluções corporativas. Por sua vez, o uso de uma metodologia definida fortalece a missão da STI de "Prover e manter soluções de Tecnologia da Informação para cumprimento da missão institucional".

Nessa conjuntura, a gestão das demandas de soluções de TI já se encontra instituída através da Resolução TRE/PI nº 320/2015, a qual regulamenta o **Plano de Acompanhamento de Desenvolvimento de Sistemas (PADS).** Todas as solicitações das unidades são submetidas ao **Comitê Diretivo de Tecnologia da Informação (CDTI)** para priorização de acordo com os critérios previstos na resolução, entre eles o alinhamento à estratégia institucional.

Assim, para atender às recomendações legais foi apresentado, em 2016, o **Manual do Processo de Software do TRE-PI (MPS-TRE/PI)**, na sua primeira versão, com o propósito de especificar e normatizar as regras que nortearão o desenvolvimento de soluções de TI.

Já em 2025, é realizada uma nova revisão aproveitando a arquitetura proposta em 2022, preservando a metodologia Kanban como referência, sendo detalhado o uso de Integração e Entrega Contínuas no desenvolvimento de software, além do alinhamento da metodologia à **Portaria Presidência n° 158/2023 TRE/PRESI/DG/ASSDG**, que trata de regras e procedimentos para desenvolvimento seguro de software no TRE-PI.

Figura 01– Evolução MPS para a versão 4

1. Introdução

Tanto a elaboração como a revisão deste documento foi realizada pela **Secretaria de Tecnologia da Informação(STI),** através da **Seção de Desenvolvimento de Soluções Corporativas (SEDESC) e da Coordenadoria de Desenvolvimento e Infraestrutura(CODIN),** observando as melhores práticas de engenharia de software utilizadas no mercado e também instituídas no **Tribunal Superior Eleitoral e outros Tribunais Regionais Eleitorais**, no caso, a **Metodologia Ágil, padrões de segurança no desenvolvimento de sistemas**, integração de equipes, aliadas à experiência da equipe de desenvolvimento do TRE-PI.

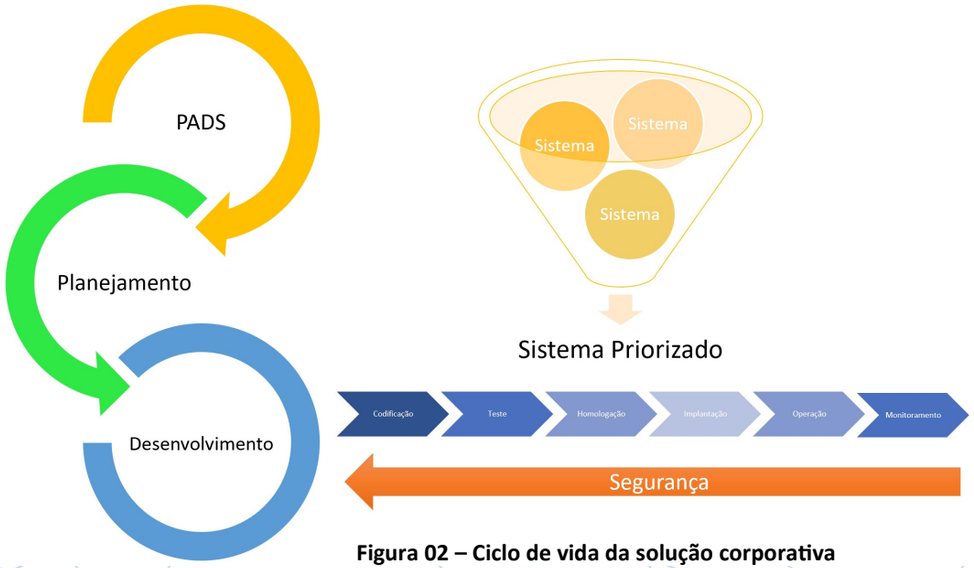


Figura 02 – Ciclo de vida da solução corporativa

Conforme apresentado na figura 02, neste processo serão detalhadas todas as etapas do ciclo de vida de uma solução corporativa. Desde o momento em que a mesma foi escalonada para atendimento, passando pelo processo de planejamento e desenvolvimento, gestão e, por fim, o momento em que não seja mais necessária sua utilização. São delineados e descritos os fluxos dos processos e subprocessos executados, os papéis e responsabilidades dos envolvidos, os recursos necessários e os resultados obtidos em cada uma das etapas.

Considera-se “Desenvolvimento” como sendo, não necessariamente, uma nova implementação, mas qualquer esforço que a equipe de desenvolvimento execute para atender a solicitação de uma unidade administrativa do TRE/PI, seja através de **estudos de viabilidade**, **análises de impactos**, **codificação**, **elaboração de documentos** e **realização de reuniões.** Como “Solução Corporativa”, considera-se um **novo software**, a **adaptação ou melhoria de um software já existente e implantado no TRE/PI ou a implantação de software de terceiros**.

2. Ciclo de vida do sistema

O ciclo de vida de um sistema corresponde às etapas pelas quais um software passa desde a sua idealização até entrar em desuso, que são: **Concepção**, **Desenvolvimento**, **Sustentação** e **Declínio**. Cada uma dessas etapas possui processos e atividades bem definidos. O fluxo apresentado na Figura 03 demonstra o macroprocesso das principais fases do ciclo de vida de um sistema.

A **etapa de concepção** corresponde ao nascimento do sistema e está descrita no capítulo 3 deste manual. Nessa fase tem-se a criação e detalhamento do escopo e requisitos da solução que deverá ser desenvolvida. Em outras palavras, é a fase em que **a ideia do cliente é traduzida em uma série de requisitos de software documentados e aptos a serem desenvolvidos**.

Figura 03 – Visão Geral do Ciclo de Vida de um sistema no TRE-PI

A **fase de desenvolvimento** refere-se à codificação, testes e implantação do sistema, processos detalhados no capítulo 4. Ainda no capítulo 4, é definida a Metodologia de Desenvolvimento de Software, aos moldes do Kanban.

A **fase de sustentação**, também conhecida como maturidade, é detalhada no capítulo 5. Refere-se à utilização plena do sistema após sua implantação em ambiente de produção e respectivas manutenções corretivas e evolutivas.

O **declínio**, definido no capítulo 6, corresponde ao momento em que o sistema passa a não mais atender o seu propósito, seja por obsolescência tecnológica, imposições legais ou decisões estratégicas de negócio. O declínio pode ter como resultado a simples desativação do sistema ou o início de um novo processo de desenvolvimento de sistemas para substituição por uma nova solução.

3. O Processo de Concepção do Sistema

3.1 Introdução

O Tribunal Regional Eleitoral do Piauí apresenta uma grande demanda quanto ao desenvolvimento de sistemas que facilitem os processos de trabalho das suas unidades administrativas. A Secretaria de Tecnologia da Informação é a unidade responsável pelo atendimento dessas demandas, sendo necessária, no entanto, a definição de um mecanismo que permita objetivamente a priorização da ordem de realização dos projetos.

O TRE-PI instituiu o Plano de Acompanhamento de Desenvolvimento de Sistemas (PADS) por meio da Resolução TRE nº 320/2015, que disciplina, no âmbito da Justiça Eleitoral do Piauí, os critérios para apresentação e priorização de solicitação de desenvolvimento de soluções corporativas.

No processo de concepção de sistemas as demandas de desenvolvimento são captadas através da atividade **Solicitar Desenvolvimento**, que segue para a **Análise Preliminar** e desencadeia em uma terceira atividade, **Priorizar Desenvolvimento**. Ainda na concepção, ocorre a **definição da arquitetura do sistema** e as atividades iniciais dos processos de **gerenciamento de requisitos** e do **escopo**.

Na figura a seguir, tem-se a ilustração das atividades e subprocessos que compõem a fase de concepção de um sistema:

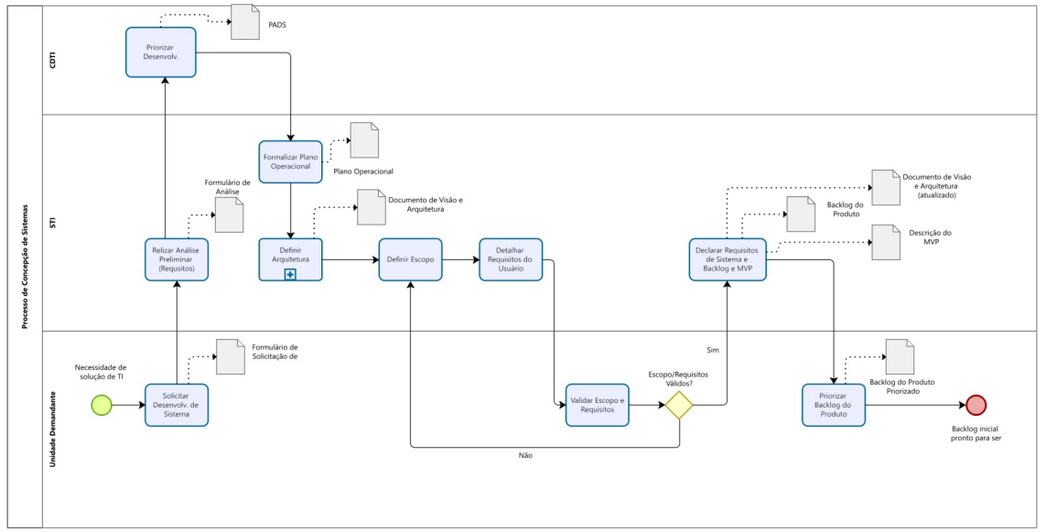


Figura 04 – Processo de Concepção de Sistemas

3.2 Solicitar o Desenvolvimento do Sistema

Representa a solicitação formal à Secretaria de Tecnologia da Informação – STI de uma demanda referente ao desenvolvimento ou à evolução de um sistema, através do Formulário de Solicitação de Sistemas, cujo modelo é definido pela resolução do PADS. Nessa atividade, a unidade demandante descreve de forma sucinta as **informações iniciais** e essenciais a respeito do sistema que está sendo proposto, justificando **a sua necessidade**, indicando **os objetivos da estratégia institucional** que serão atendidos e **o gestor responsável** pelos procedimentos que serão informatizados.

O pedido é feito através do sistema eletrônico de tramitação dos processos administrativos utilizado no TRE/PI e obedece a um prazo estabelecido na Resolução do PADS. Essas informações são fundamentais para a realização da etapa seguinte, onde é executada a **Análise Preliminar**.

3.3 Realizar Análise Preliminar

Durante essa etapa, pretende-se obter de forma mais detalhada um entendimento da demanda. Ao receber a solicitação formalizada, o gestor responsável pelo desenvolvimento de sistemas escolhe um técnico da sua equipe para coletar, através de entrevistas, mais informações junto ao gestor indicado no Formulário de Solicitação de Sistemas pela unidade demandante.

O técnico responsável por essa análise, após ter uma visão clara do sistema, expõe no Formulário de Análise Preliminar, cujo modelo é definido pela resolução do PADS, as **partes interessadas**, **escopo** e **requisitos iniciais**, além de uma **estimativa de prazos**.

As análises preliminares servirão de base para que o Comitê Diretivo de Tecnologia da Informação – CDTI determine uma **priorização de desenvolvimento de sistemas**, etapa descrita a seguir, de acordo com os critérios previstos na Resolução do PADS.

3.4 Priorizar o Desenvolvimento

Corresponde à fase de priorização das demandas de acordo com os critérios estabelecidos na resolução do PADS. Para isso, os membros do Comitê Diretivo de Tecnologia da Informação-CDTI se reúnem ordinariamente, em períodos definidos pela resolução do PADS, discutindo as solicitações feitas através dos formulários de análise preliminar.

O resultado dessa priorização constituirá uma ordem a qual a unidade responsável pelo desenvolvimento obedecerá para escalonar a sequência de atendimento. O **PADS será apreciado pela Presidência do TRE/PI para aprovação através de portaria**.

As demandas não previstas para a última reunião ordinária do CDTI serão solicitadas utilizando o mesmo Formulário de Solicitação de Sistemas citado no item 2.2.1. A análise preliminar será realizada e apreciada na próxima reunião ordinária do CDTI. Para analisar as solicitações urgentes, poderá ser convocada uma reunião extraordinária por determinação da Presidência do TRE/PI.

3.5 Formalizar Plano Operacional

Cada um dos Sistemas componentes do PADS será considerado um Projeto. Logo, a STI, na figura do Gestor de Desenvolvimento, irá formalizar um Plano Operacional, o qual seguirá os trâmites previstos em normas internas.

3.6 Definir a Arquitetura

A arquitetura do sistema é **o conjunto de soluções tecnológicas** que, estabelecido de maneira integrada e coordenada, **provê os recursos necessários para que o software desempenhe seu propósito**. A definição dessa arquitetura envolve a análise, caso a caso, do tipo de solução a ser desenvolvida, requisitos funcionais e não funcionais existentes, entre outras variáveis aplicáveis ao caso concreto.

Essa definição envolve o time de desenvolvimento, que fará a **sugestão inicial**, e o departamento de desenvolvimento, que terá decisão final sobre a aprovação da arquitetura a ser utilizada. O resultado desse processo é o **documento de declaração da visão e arquitetura**, que pode vir a ser atualizado conforme o projeto avança. O processo de definição da arquitetura é ilustrado no diagrama a seguir:

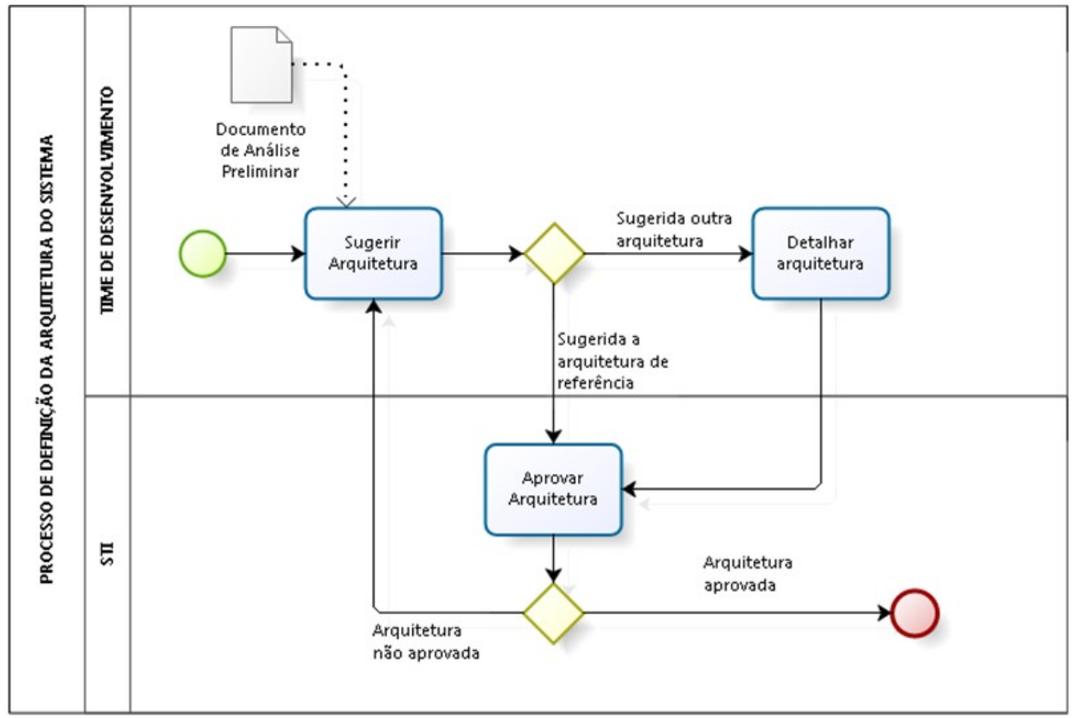


Figura 05 – Processo de Definição da Arquitetura

3.6.1 Sugerir a Arquitetura

O Time de Desenvolvimento, de posse do **Documento de Análise Preliminar**, fará um estudo da solução a ser implementada e fará a sugestão do modelo arquitetural a ser utilizado. Sempre que possível, o Time deverá optar por adotar o **modelo arquitetural** sugerido no item 3.5.4, de modo a favorecer a padronização e a maximização no uso dos conhecimentos já consolidados pelo quadro de colaboradores da STI.

A sugestão da arquitetura a ser utilizada, deve ser encaminhada à chefia da seção de desenvolvimento de software.

3.6.2 Detalhar a Arquitetura

Esta atividade é ativada quando o Time de Desenvolvimento opta por sugerir o uso de um **modelo arquitetural diverso** do proposto no item 3.5.4. Nesses casos, deve ser preenchido o Formulário de Arquitetura Detalhado, e o mesmo deve ser submetido à aprovação da chefia da seção de desenvolvimento de software.

3.6.3 Aprovar a Arquitetura

Nesta atividade a chefia da seção de desenvolvimento de software, **juntamente com Núcleo de Cibersegurança**, analisam o projeto a ser desenvolvido e manifesta-se sobre a adoção ou não do modelo arquitetural proposto pela equipe de desenvolvimento.

3.6.4 Arquitetura de Referência

Na versão 1.0 do Manual de Processo de Software do TRE-PI, a arquitetura padrão era baseada em um modelo de arquitetura monolítico, onde todos os elementos e requisitos do sistema a ser desenvolvido eram implementados e consolidados em um único aplicativo executável.

Apesar do uso de boas práticas de padrões de projeto e aliadas a tecnologias coorporativas, como Java EE, o uso desse padrão arquitetural, ao longo do tempo, começou a apresentar desafios para a equipe de desenvolvimento, como por exemplo:

- Aumento da complexidade e tamanho do sistema
- Alta dependência de componentes de código
- Dificuldade de escalabilidade
- Pouca flexibilidade para sistemas mais complexos
- Dificuldade para colocar as alterações em produção

Para a **versão 2.0 do MPS** foi estabelecido pela equipe de desenvolvimento do TRE-PI o uso da arquitetura de microsserviços, e nesta **versão (3.0)** se mantém.

A arquitetura de microsserviços consiste em uma coleção de pequenos serviços autônomos. Cada serviço é independente e deve implementar uma única funcionalidade do negócio dentro de um contexto limitado. Um contexto limitado é uma divisão natural de uma regra de negócio e fornece um limite explícito dentro do qual um modelo de domínio existe.

A figura a seguir apresenta o esquema da arquitetura de referência utilizando a abordagem por microsserviços.

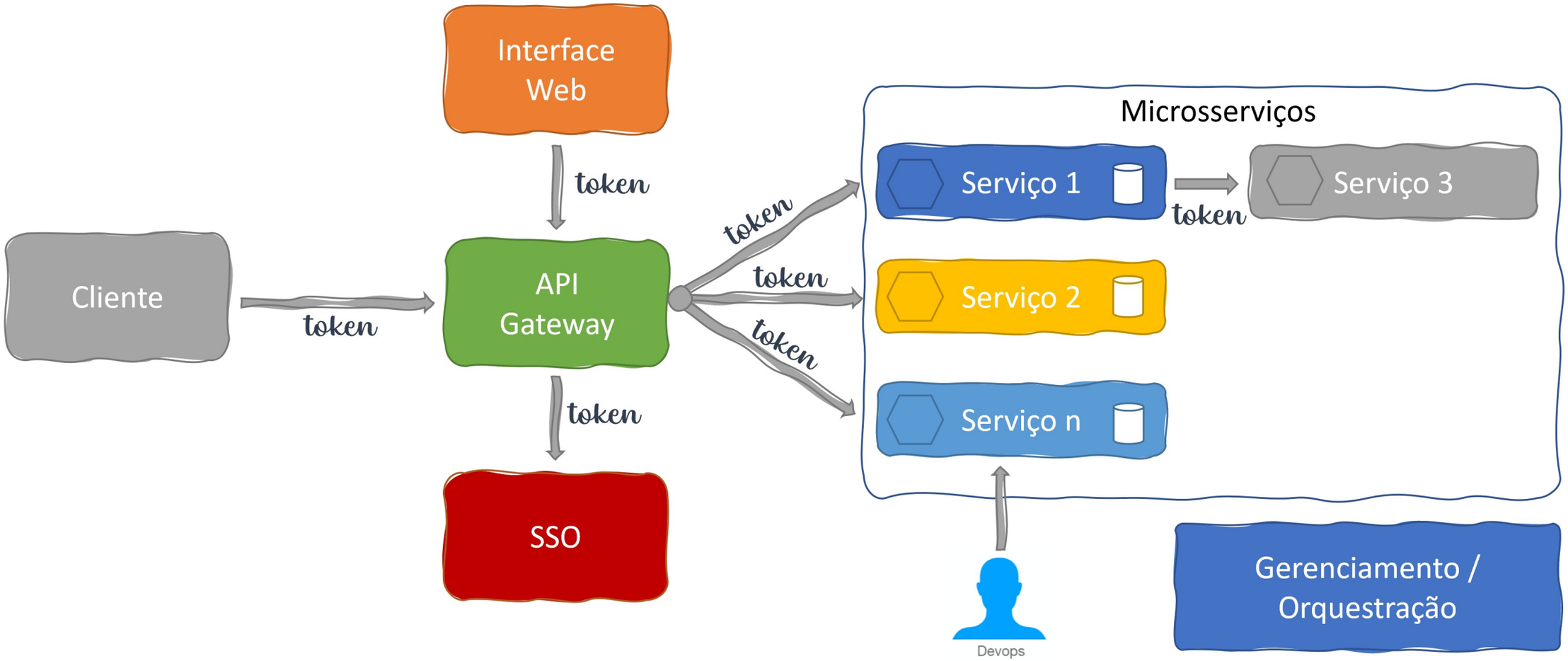


Figura 06 – Arquitetura de Referência

3.6.4.1 Microserviços

Os microsserviços consistem em uma abordagem arquitetônica na qual um único aplicativo é composto de muitos componentes ou serviços menores que são implementáveis de forma independente e possuem as seguintes características:

- São pequenos, independentes e fracamente acoplados.
- Os serviços podem ser implantados de maneira independente.
- Cada serviço possui uma base de código em separado, sendo responsáveis por manter os seus próprio dados.
- Os serviços comunicam-se entre si por meio de APIs RESTful bem definidas.
- Podem ser implementados em tecnologias distintas, não precisando compartilhar a mesma pilha de tecnologias, bibliotecas ou estruturas.

3.6.4.2 Padrão API Gateway

O gateway de API é o ponto de entrada para os clientes. Em vez de chamar os serviços diretamente, os clientes chamam a API Gateway, que encaminha a chamada para os serviços adequados.

As vantagens de usar uma API Gateway incluem:

- Desacoplar os clientes dos serviços.
- Utilização de outros protocolos de mensagens e não somente o HTTP.
- Executar outras funções abrangentes, como autenticação, registro em log, terminação SSL e balanceamento de carga.

3.6.4.3 Gerenciamento / Orquestração

Esse componente é responsável por colocar serviços em nós—também conhecidos como “Containers”, identificar falhas, balancear a carga entre os serviços e assim por diante. Normalmente, esse componente é uma tecnologia pronta para uso, como Kubernetes, em vez de algo criado de maneira personalizada.

Com o gerenciamento e a orquestração dos microsserviços, implantados em nós do tipo “Container”, pode-se automatizar e gerenciar tarefas como:

- Provisionamento e implantação
- Configuração da aplicação com base no container em que ela será executada
- Alocação de recursos
- Escalabilidade e disponibilidade dos containers
- Balanceamento de carga e roteamento de tráfego
- Monitoramento
- Proteção das interações entre os serviços

3.6.4.4 SSO—Single Sign-On

Single sign-on (SSO) é um esquema de autenticação que permite que um usuário efetue login com um único ID em qualquer um dos vários sistemas de software relacionados, mas independentes.

Em uma arquitetura de microsserviço, o cliente, seja um usuário ou um serviço, interage com uma coleção de outros serviços, cada um com uma necessidade potencial de saber quem é o usuário/cliente. Assim, um serviço centralizado de autenticação e autorização permite armazenar as credenciais usadas para a autenticação inicial e convertê-las nas credencias necessárias para os diversos serviços desenvolvidos ou em uso pelo TRE-PI.

Os benefícios do uso dessa abordagem incluem:

- Redução do risco de acesso por site de terceiros—autenticação federada
- Redução do excesso de entradas de senhas por meio da combinação usuário/senha
- Administração simplificada de autenticação e autorização a recursos
- Melhor controle administrativo

No MPS versão 3.0, o SSO na arquitetura de referência irá adicionar uma camada extra de segurança a API Gateway, aumentando a visibilidade de autorização e roteamento e diminuindo a quantidade de API expostas, garantindo a diminuição das superfícies de ataques aos sistemas.

3.6.4.5 Interface Web

Essa é a camada de apresentação do sistema, ela apresenta a interface do usuário e de comunicação do aplicativo. Seu principal objetivo é exibir as informações e coletar os dados dos usuários. Essa camada será desenvolvida utilizando as tecnologias web, como html, css e Javascript.

3.6.4.6 DevSecOps

DevSecOps significa desenvolvimento, segurança e operações. É uma abordagem à cultura, automação e design da plataforma que integra segurança como uma responsabilidade compartilhada em todo o ciclo de vida de um produto.

O uso dessa abordagem implicará em uma evolução natural e necessária na forma como a equipe de desenvolvimento do TRE-PI abordará as questões relativas à segurança em todo o processo de desenvolvimento e manutenção de software.

3.6.4.6.1 Integração e Entrega Contínua (CI/CD)

Este procedimento, conhecido como **Pipeline de Integração Contínua, Entrega Contínua e/ou Implantação Contínua (CI/CD)**, é um dos pilares essenciais para a efetiva implementação da cultura e práticas **DevSecOps**. Consiste fundamentalmente em utilizar um conjunto robusto de ferramentas e automações para otimizar e proteger o ciclo de vida de desenvolvimento de software. O objetivo primordial é automatizar as etapas de **integração de código, construção da aplicação, execução de testes (funcionais, de segurança e de qualidade) e a implantação** das novas versões do software. Esta automação é projetada para ocorrer de forma frequente e com mínima intervenção humana, resultando em lançamentos mais rápidos, consistentes, seguros e confiáveis.

O processo automatizado, que integra a segurança desde as fases iniciais (Shift Left Security), geralmente se desenrola através dos seguintes passos interconectados:

Versionamento e Disparo do Pipeline

O desenvolvedor consolida suas alterações de código e as envia (através de um commit e push) para um **servidor de controle de versão centralizado**, como o GitLab.

Essa ação de enviar o código ao repositório principal ou a uma *branch* específica serve como o **gatilho (trigger)** que inicia automaticamente o pipeline de CI/CD.

Construção, Testes Automatizados e Análise de Segurança

Uma vez que o novo código é recebido pelo servidor de versionamento, uma **rotina automatizada (o pipeline de CI/CD)** entra em ação, executando uma sequência de tarefas críticas:

- **Build (Construção):** O código fonte é compilado e empacotado, gerando um artefato de software executável.
- **Testes Automatizados:** Diversos níveis de testes são executados, como testes unitários (para verificar componentes individuais), testes de integração (para verificar a interação entre componentes) e, possivelmente, testes de aceitação.
- **Análise Estática de Qualidade de Código:** Ferramentas analisam o código em busca de "code smells" (maus odores de código), complexidade ciclomática elevada, duplicação de código e aderência a padrões de estilo.

- **Verificação Estática de Vulnerabilidades de Segurança (SAST):** Ferramentas de *Static Application Security Testing* (SAST), como o Fortify, Inspecionam o código-fonte, bytecode ou código binário em busca de padrões que indiquem vulnerabilidades de segurança conhecidas (ex: SQL Injection, Cross-Site Scripting, etc.) antes mesmo da aplicação ser executada. Esta é uma etapa crucial do "Sec" em DevSecOps.

Decisão e Implantação Automatizada (Entrega ou Implantação Contínua): Após a execução de todas as verificações anteriores:

- **Validação:** O pipeline avalia os resultados. Se o código passar em todos os testes (funcionais e de qualidade) e, crucialmente, não apresentar vulnerabilidades de segurança críticas ou que violem as políticas estabelecidas, ele é considerado apto para a próxima fase.
- **Implantação:** Conforme os parâmetros configurados no pipeline (que podem variar de acordo com a *branch*, *tags* ou gatilhos manuais), o artefato de software aprovado é automaticamente implantado. Isso pode ocorrer em diferentes ambientes:
    - **Ambiente de Desenvolvimento:** Para testes exploratórios e integração contínua com outras funcionalidades em desenvolvimento.
    - **Ambiente de Homologação (Staging/QA):** Uma réplica do ambiente de produção, onde são realizados testes de aceitação do usuário (UAT), testes de carga, e validações finais antes da produção.
    - **Ambiente de Produção:** Onde a aplicação fica disponível para os usuários finais. A implantação direta em produção após todos os checks caracteriza a **Implantação Contínua**. Se houver uma aprovação manual antes da produção, mesmo com o processo automatizado até este ponto, caracteriza-se como **Entrega Contínua**.

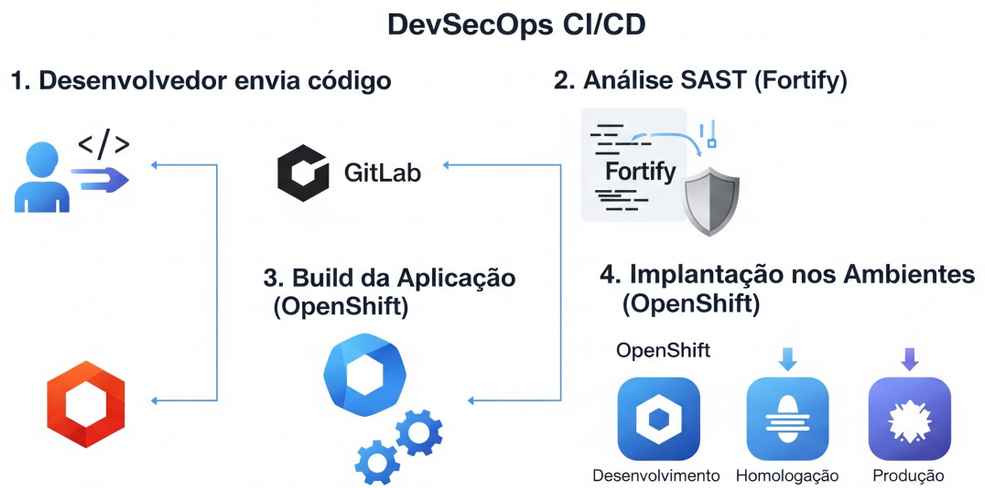


Figura 07—Ilustração do Processo CI/CD

3.6.4.6.2 Desenvolvimento Seguro de Software

O segurança no desenvolvimento de sistemas já é contemplado na cultura DEVSECOPS, sendo que o TRE-PI normatizou na Portaria Presidência n° 158/2023 uma série de procedimentos para atingir esse objetivo, considerando os seguintes tópicos:

3.6.4.6.3 Análise de Vulnerabilidades

Juntamente com a equipe de Segurança da Informação, a equipe de desenvolvimento deverá realizar o tratamento de vulnerabilidades do processo de software, tais como:

1. Recebimento de notificação de vulnerabilidade
2. Classificação do Risco da vulnerabilidade
3. Notificação da correção da vulnerabilidade
4. E análise da causa raiz da vulnerabilidade

3.6.4.6.4 Boas práticas recomendadas para proteção de dados pessoais

Algumas práticas devem ser adotadas, no processo de concepção e desenvolvimento das aplicações, obedecendo a padrões estabelecidos na indústria de desenvolvimento de software, tais como:

I - **Privacy By Design**: assegura a proteção de dados pessoais deverá ser estabelecida desde a concepção do software ou componentes compreendendo todo o ciclo de vida, onde a equipe deverá realizar uma abordagem proativa na proteção de dados pessoais; e

II – **Privacy By Default**: o software deverá resguardar a exposição de dados pessoais salvaguardando a privacidade, sendo o mais restritivo possível tanto na exposição/visualização de dados pessoais quanto na coleta;

III—As vulnerabilidades que incidem sobre dados pessoais devem ser tratados com prioridade sobre qualquer outro tipo;

IV - Qualquer componente de terceiros que manuseie dados pessoais devem sofrer uma análise adicional, sendo inventariado e validado.

3.7 Detalhar Requisito do Usuário

Esta atividade consiste no detalhamento dos requisitos inicialmente levantados junto ao usuário na atividade Realizar Análise Preliminar. Tem-se como objetivo elaborar uma lista de requisitos, funcionalidades que o sistema deve a executar e/ou restrições sobre as quais deve operar.

3.8 Definir o Escopo

Esta atividade visa a definição dos limites do sistema. Sua execução ocorre de maneira paralela ao detalhamento dos requisitos devido à forte integração e interdependência dessas duas atividades. Alterações nos requisitos certamente terão impacto no escopo, e vice-versa. O propósito do sistema deve ficar claro na Declaração do escopo, que deverá ser explicitamente validada pela Unidade Demandante.

Essa tarefa complementa e elabora o escopo inicialmente levantado na tarefa Realizar Análise Preliminar, e pode ser ativada através de eventos das fases de Desenvolvimento e Sustentação.

3.9 Validar Escopo e Requisito

Esta atividade tem por objetivo obter, junto à unidade demandante, a confirmação de que o escopo e os requisitos apresentados de fato refletem suas expectativas quanto à solução que deve ser implementada.

3.10 Declarar Requisitos do Sistema, Backlog e MVP

É nesta atividade que a unidade responsável pelo desenvolvimento da solução fará a tradução dos requisitos validados pelo usuário em requisitos de sistema. Ter-se-á ao fim dessa tradução o artefato chamado *Backlog* do Produto, que reúne todos os requisitos do sistema escritos em linguagem técnica, de modo a subsidiar o time de desenvolvimento com uma visão abrangente do trabalho a ser feito. Uma outra saída possível, é a atualização do Documento de Visão e Arquitetura, nos casos em que os detalhamentos e consecutivas validações no escopo e requisitos venham a justificar essa atualização.

Uma parte dos requisitos do backlog irá compor um documento descrevendo o **Produto Mínimo Viável**, o qual representará um entregável possível de ser usado pelo demandante em que será avaliado a viabilidade do projeto, com atendimento das necessidades prementes do Gestor do Sistema.

3.10 Priorizar Backlog do Produto

A priorização do *backlog* do produto consiste na definição, pela unidade demandante, de quais funcionalidades do sistema têm prioridade sobre outras. Isso possibilita que a fase de desenvolvimento seja conduzida de modo a seguir essa prioridade, proporcionando a entrega artefatos de software tidos como mais importante num menor espaço de tempo, acelerando a geração de valor. Ao fim desta atividade, ter-se-á uma versão inicial do *backlog* do produto e a fase de desenvolvimento poderá ser iniciada tão logo o sistema em questão adquira prioridade no PADS.

3.11 Matriz de Responsabilidades

ATIVIDADES/PAPÉIS	GESTOR DE DESENVOLVIMENTO	GESTOR DO SISTEMA	ANALISTA DE REQUISITOS	EQUIPE
Solicitar desenvolvimento de Sistema	-	R/E	-	-
Realizar análise preliminar	R/E	I	I	I
Priorizar desenvolvimento	-	R/E	E	-
Formalizar Plano de Ação	R/E	I	-	-
Definir arquitetura	C/I	-	C/I	R/E
Detalhar requisitos de usuário	-	C/I	R/E	C/I
Definir escopo	-	R/E	E	-
Validar escopo e requisitos	-	R/E	E	-
Detalhar requisitos de sistema, backlog e MVP	R/E	C/I	C/I	C/I
Priorizar backlog do produto	-	R/E	E	-

Legenda: R – Responsável, E – Executor, C – Consultado, I – Informado.

Tabela 01 – Processo de Concepção de Sistemas

3.12 Recursos e Resultados

ATIVIDADES	RECURSOS	RESULTADOS
Solicitar desenvolvimento de Sistema	Modelo do Formulário de Solicitação de Sistema	Formulário de Solicitação de Sistema preenchido
Realizar análise preliminar	Formulário de Solicitação de Sistema preenchido	Formulário de análise preliminar
Priorizar desenvolvimento	Formulário de Análise Preliminar; Atas das reuniões com o gestor do sistema	PADS
Formalizar Plano de Ação	Formulário de Análise Preliminar Atas de reuniões com o gestor do sistema	Plano de Ação
Definir arquitetura	Formulário de Análise Preliminar.	Documento de visão e arquitetura
Detalhar requisitos de usuário	Documento de Visão e Arquitetura do Produto.	Documento de Visão e Arquitetura do Produto (Atualizado).
Definir escopo	Documento de Visão e Arquitetura do Produto.	Documento de Visão e Arquitetura do Produto (Atualizado).
Validar escopo e requisitos	Documento de Visão e Arquitetura do Produto.	Validação do Documento de Visão e Arquitetura do Produto (Atualizado).
Declarar requisitos de sistema, backlog e MVP	Documento de Visão e Arquitetura do Produto.	Documento de Visão, Arquitetura do Produto (Atualizado) e Descrição do MVP; Backlog do Produto.
Priorizar backlog do produto	Backlog do produto	Backlog do produto priorizado.

Tabela 02 – Recursos e Resultados do Processo de Concepção de Sistemas

4. O Processo de Desenvolvimento de Sistemas

4.1 Introdução

O Tribunal Regional Eleitoral do Piauí apresenta uma grande demanda quanto ao desenvolvimento de sistemas que facilitem os processos de trabalho das suas unidades administrativas. A Secretaria de Tecnologia da Informação é a unidade responsável pelo atendimento dessas demandas, sendo necessária, no entanto, a definição de um mecanismo que permita objetivamente a priorização da ordem de realização dos projetos. Uma vez priorizados, cada um dos projetos deve ser conduzido através de um processo de desenvolvimento consolidado, que seja capaz de coordenar a evolução das etapas de trabalho e, ao mesmo tempo, ágil o suficiente para absorver as mudanças que se fazem necessárias ao longo da execução de todo projeto de software.

O TRE-PI instituiu o Plano de Acompanhamento de Desenvolvimento de Sistemas (PADS) por meio da Resolução TRE nº 320/2015, que disciplina no âmbito da Justiça Eleitoral do Piauí os critérios para apresentação e priorização de solicitação de desenvolvimento de soluções corporativas. Uma vez que um projeto de software recebe a prioridade necessária para que tenha seu desenvolvimento iniciado, esse trabalho deve ser realizado observando o Processo de Desenvolvimento definido neste capítulo.

4.2 Metodologia de Desenvolvimento de Sistemas

O desenvolvimento de software de maneira indisciplinada eleva as chances de retrabalho, desperdício, insegurança e frustração para a unidade demandante de um sistema. Em razão disso, é imprescindível definir e seguir padrões preestabelecidos que norteiem o fluxo de construção de um programa, considerando as melhores práticas para modelagem, programação, métrica, arquitetura e testes.

A institucionalização de uma metodologia padronizada fortalece a missão da Secretaria de Tecnologia da Informação - STI de “Prover e manter soluções de Tecnologia da Informação para cumprimento da missão institucional”, além de atender as boas práticas de Governança de TI.

4.2.1 Metodologia Ágil

Entre as melhores práticas em termos de processo de desenvolvimento destaca-se a Metodologia de Desenvolvimento Ágil, cujos princípios foram relacionados em 2001, quando profissionais veteranos da área de construção de software criaram o **Manifesto para o Desenvolvimento Ágil de Sistemas**, chamado apenas de **Manifesto Ágil**, o qual descrevia abordagens de desenvolvimento que passassem a valorizar:

Indivíduos e interação entre eles mais que processos e ferramentas;

Sistema em funcionamento mais que documentação abrangente;

Colaboração com o cliente mais que negociação de contratos;

Responder a mudanças mais que seguir um plano.

Observa-se que, mesmo havendo valor nos itens à direita, valoriza-se mais os itens à esquerda.

O Manifesto Ágil tem seus princípios baseados no *Extreme Programming (XP)*, uma metodologia de desenvolvimento de software, nascida nos Estados Unidos ao final da década de 90, a qual propõe criar sistemas de melhor qualidade, produzidos em menos tempo e de forma mais econômica que o habitual. Tais objetivos são alcançados através de um pequeno conjunto de **valores, princípios e práticas,** que diferem bastante da forma tradicional de desenvolver software.

O Manifesto Ágil recomenda ainda 12 princípios:

• Nossa maior prioridade é satisfazer o cliente através da entrega adiantada e contínua de software de valor;

• Aceitar mudanças de requisitos mesmo no fim do desenvolvimento. Processos ágeis se adequam a mudanças para que o cliente possa tirar vantagens competitivas;

• Entregar software funcionando com frequência, na escala de semanas até meses, com preferência aos períodos mais curtos;

• Pessoas relacionadas a negócios e desenvolvedores devem trabalhar em conjunto e diariamente, durante todo o curso do projeto;

• Construir projetos ao redor de indivíduos motivados. Dando a eles o ambiente e suporte necessário, e confiar que farão seu trabalho;

• O Método mais eficiente e eficaz de transmitir informações para, e por dentro de um time de desenvolvimento, é através de uma conversa cara a cara;

• Software funcional é a medida primária de progresso;

• Processos ágeis promovem um ambiente sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter indefinidamente passos constantes;

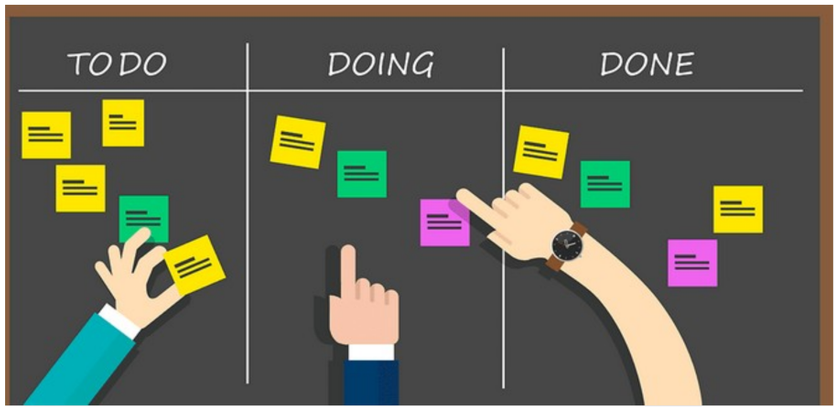
• Contínua atenção à excelência técnica e bom design aumenta a agilidade;

• Simplicidade: a arte de maximizar a quantidade de trabalho que não precisou ser feito;

• As melhores arquiteturas, requisitos e designs emergem de times auto-organizáveis;

Em intervalos regulares, o time reflete em como ficar mais efetivo, então se ajustam e otimizam seu comportamento.

4.2.2 Kanban



Criada pela Toyota, na década de 40, o Kanban foi criado com o objetivo de suplantar a indústria automobilística norte-americana em termos de produtividade. Hoje é utilizada em qualquer tipo de projeto, inclusive no desenvolvimento de Softwares.

Na verdade, o Kanban constitui um framework, o qual pode ser usado para dar origem a qualquer tipo de metodologia de produção de um produto. No caso em específico, este manual o utilizará para o processo de desenvolvimento de sistemas informatizados.

As funcionalidades do sistema são dispostas em um quadro, físico ou eletrônico, com a designação de quem irá realizá-la. Este quadro é divido em colunas com os estágios mínimos de “a fazer”, “fazendo” e “feito”, podendo-se atribuir mais estágios a critério da equipe.

Ao contrário da metodologia tradicional de desenvolvimento de sistemas, que considera que o produto só tem valor para o usuário quando é entregue em sua totalidade, a metodologia ágil propõe frequentes entregas de partes do produto que agreguem valor ao usuário.

O Kanban pressupõe que as funcionalidades, quando agrupadas, e constituir algo possível de entrega, a equipe avalia, juntamente com o demandante, e uma parte do produto poderá ser entregue ao cliente.

4.2.2.1 Termos Importantes

Alguns termos são necessários compreender para seguir com o entendimento deste manual:

Backlog: é uma visão nocional do local que mostra uma lista consolidada de itens de trabalho (Histórias de usuários/defeitos/problemas) que devem ser trabalhados. Pode-se pesquisar através do Backlog e adicionar cartões ao Software de Quadro Kanban.

Quadro Kanban: é uma das ferramentas para implementar o método Kanban. O quadro é dividido em, no mínimo, 3 colunas – Para Fazer, Fazendo, Feito, representando as etapas de um processo.

Raia: são divisões horizontais de um quadro Kanban, que ajudam a otimizar o fluxo de trabalho. As colunas representam etapas e as raias categorizam o trabalho. Raias podem ser usadas para representar times, classes de serviço, prioridade, etc.

Cartão kanban: representam visualmente os itens de trabalho. Cada cartão é uma tarefa, que se move pelas colunas do quadro Kanban. Os cartões contêm informações sobre o item de trabalho. Eles não possuem diferença em tamanho, porque a ideia é dividir um projeto em pequenas tarefas que podem ser concluídas rapidamente.

WIP (Work in Progress): ou Trabalho em Progresso, é a quantidade de trabalho sendo iniciada, independentemente da sua subcoluna atual.

Limites de WIP: é uma estratégia para evitar a sobrecarga e a mudança de contexto enquanto focamos em coisas importantes. A aplicação de limites de WIP é a segunda prática central do Kanban e assegura um fluxo saudável.

4.2.3 Objetivo do Manual do Processo de Software

O Manual do Processo de Software do TRE/PI (MPS-TRE/PI) visa, dentre outros fins, determinar padrões para o desenvolvimento de sistemas no âmbito deste Regional, seguindo e incorporando os parâmetros da metodologia ágil, definindo etapas, papéis e artefatos necessários para a correta realização dos processos de gerenciamento, desenvolvimento e manutenção de sistemas, bem como na implantação de soluções desenvolvidas por outros órgãos do Poder Judiciário. Visa-se portanto, prover à equipe técnica uma maior efetividade nas suas atividades e às unidades do TRE/PI maior satisfação quanto ao atendimento de soluções de seu interesse.

4.2.4 Papéis e Responsabilidades

Os papéis e respectivas responsabilidades dos envolvidos na fase de Desenvolvimento do MPS são descritos neste tópico.

4.2.4.1 Gestor de Desenvolvimento de Sistema

Responsável por gerenciar o processo de atendimento das demandas referentes ao desenvolvimento de sistemas no TRE/PI. São suas responsabilidades:

- Garantir o uso das recomendações descritas nesta metodologia;
- Designar analistas de requisitos para realização de análises preliminares;
- Gerenciar a realização das Análises Preliminares;
- Acompanhar a execução dos cronogramas definidos;
- Reconhecer perfis profissionais para alocá-los nas atividades adequadas;
- Definir as equipes de desenvolvimento;
- Garantir a integração entre a equipe de desenvolvimento e o gestor do sistema.
- Controlar as atividades de mudanças de requisitos e a entrada de novos requisitos;
- Gerenciar os fatores de risco quanto ao atendimento de mudanças nos requisitos;
- Fazer a mediação entre o gestor do sistema e a equipe de desenvolvimento;
- Aceitar ou rejeitar o que for entregue nas *sprints;*
- Acompanhar a homologação de documentos junto ao gestor do sistema;
- Modificar o *backlog* do produto;

4.2.3.2 Gestor do Sistema

Pessoa indicada pela unidade demandante de um sistema para acompanhar e fornecer todas as informações necessárias para a unidade de desenvolvimento. É responsável por:

- Fornecer as informações necessárias em todas as etapas de desenvolvimento;
- Participar de reuniões acordadas;
- Acompanhar a execução das atividades junto à unidade de desenvolvimento;
- Validar os documentos de requisitos gerados;
- Validar os produtos elaborados na fase de desenvolvimento;
- Homologar os produtos finais.

4.2.3.3 Analista de Requisitos

Técnico designado para realizar a análise preliminar de um sistema, coletando junto ao gestor do sistema os dados indispensáveis para o reconhecimento inicial do problema que a unidade demandante precisa solucionar. A ele compete:

- Acordar reuniões com o gestor do sistema para captar informações e dirimir dúvidas;
- Elaborar o documento "Formulário de Análise Preliminar".

4.2.3.4 Equipe de Desenvolvimento

Grupo de pessoas responsáveis pelo desenvolvimento de um sistema. Também denominado Equipe de Desenvolvimento. De acordo com o Método Ágil, possuem habilidades multidisciplinares para que sejam capazes de se auto-organizarem durante o projeto. Não há uma divisão funcional através de papéis tradicionais, tais como programador, *designer* ou analista de testes. Os integrantes do projeto trabalham juntos para entregar das funcionalidades ou sistema. A equipe é responsável por:

- Seguir as diretivas fundamentadas no MPS;
- Codificar em linguagem de programação a solução de sistema, de acordo com as regras do negócio, os requisitos e documentos de apoio;
- Tomar decisões técnicas que orientem o design e a implementação do projeto;
- Produzir e validar a documentação a ser entregue;
- Definir a arquitetura dos sistemas;
- Prezar pela qualidade do código-fonte, documentando a codificação para futuras manutenções;
- Analisar e modelar o banco de dados de desenvolvimento e testes;
- Criar as interfaces solicitadas pelo usuário;
- Inspecionar código de outros desenvolvedores;
- Elaborar o manual do usuário, quando necessário;
- Propor soluções ou alternativas para problemas;
- Identificar e construir casos e modelos de teste, segundo os testes de aceitação derivados das histórias dos usuários;
- Realizar testes automatizados das unidades implementadas e das interfaces;
- Realizar testes de integração, de sistema e de aceitação em ambiente de homologação similar ao ambiente de produção.

4.2.3.5 Equipe de Infraestrutura

Equipe responsável por manter disponíveis os serviços de TI instalados nos servidores do Regional. Participam do processo aqui definido sempre que a equipe de desenvolvimento precisar que sejam instalados e configurados programas e bancos de dados nos equipamentos servidores. Compete a essa equipe:

- Criar e manter a estrutura do banco de dados de homologação e produção;
- Implantar o produto em ambiente de homologação e produção;
- Zelar pela integridade e desempenho da estrutura de banco de dados.

4.3 Processo de Desenvolvimento

O macroprocesso de Desenvolvimento de Software é ilustrado na figura 6 e posteriormente detalhado nas seções a seguir.

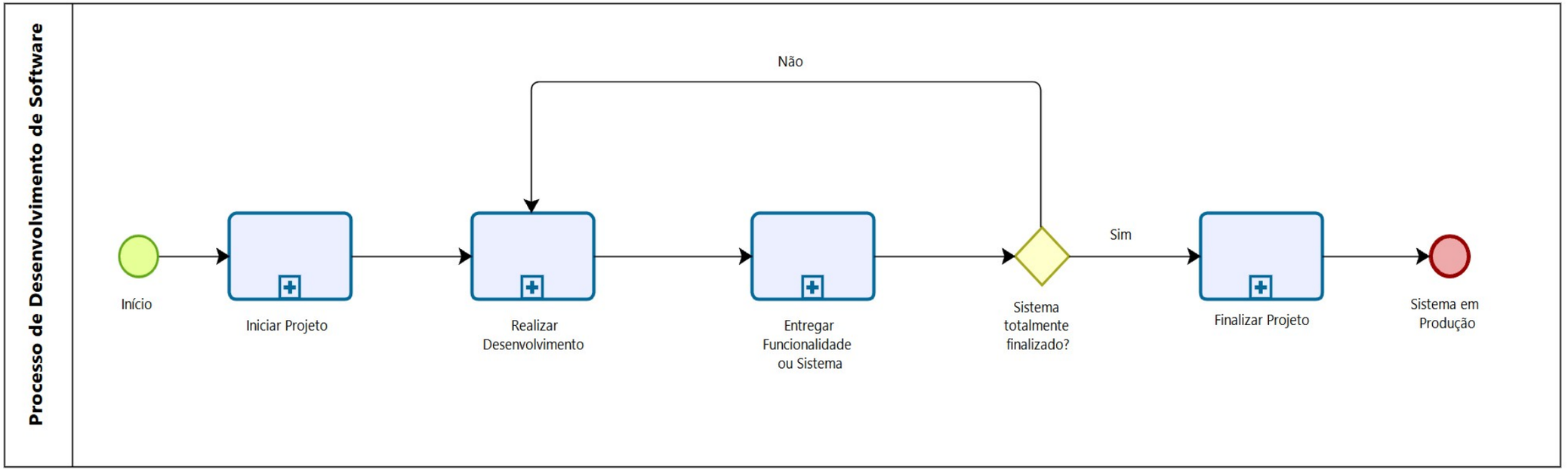


Figura 08 – Etapas do Desenvolvimento de Sistemas no TRE-PI

4.3.1 Iniciar Sistema

O início do projeto envolve as atividades, que são ilustradas no diagrama abaixo e detalhadas nos próximos tópicos.

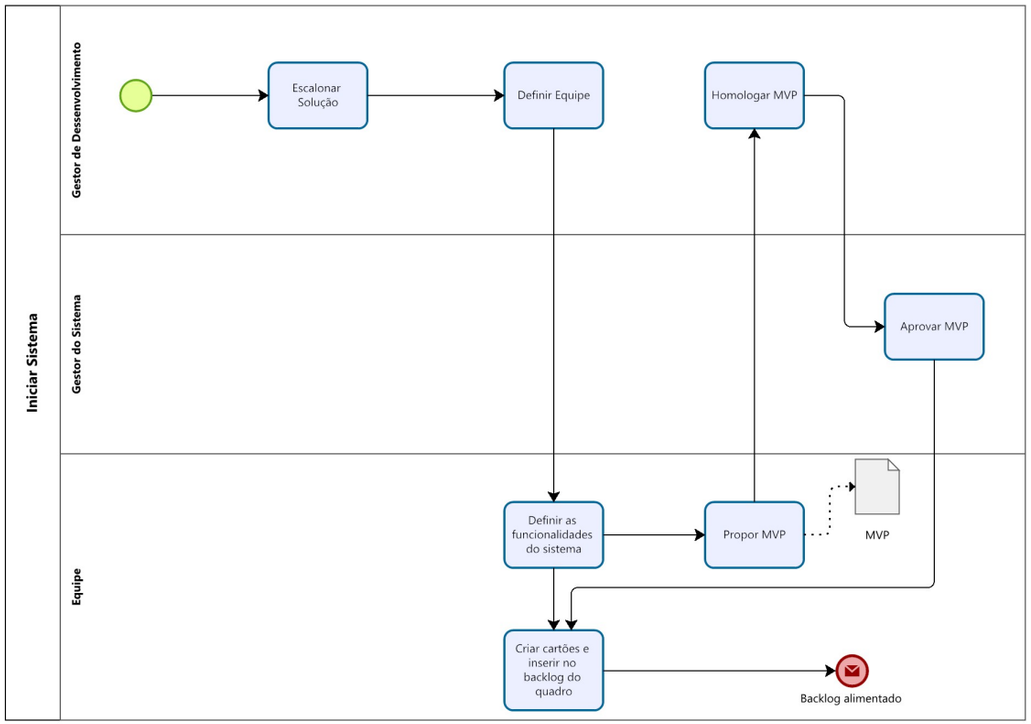


Figura 09 – Fluxo do Subprocesso Iniciar Projeto

4.3.1.1 Escalonar Solução

Nesta fase, o gestor de desenvolvimento de sistemas seleciona o próximo projeto a ser atendido, seguindo a sequência estabelecida na Portaria do PADS, resultante da última priorização do CDTI. Havendo disponibilidade de equipes, várias soluções podem ser escalonadas simultaneamente.

No caso de ocorrer algum impedimento para que um sistema seja escalonado em razão de premissas não atendidas, a demanda ficará pendente e o próximo sistema será escolhido no intuito de não haver ociosidade da equipe. Uma vez resolvida a pendência, será analisado se alguma das soluções em andamento poderá ser interrompida. Caso contrário, a demanda permanecerá aguardando uma equipe ser liberada.

4.3.1.2 Definir Equipe

Corresponde à montagem da equipe pelo **Gestor de Desenvolvimento de Sistemas**. O grupo terá perfil para o atendimento da demanda indicada e será composto por profissionais que exercerão os seguintes papéis: Analista de Requisitos e Equipe de desenvolvimento.

4.3.1.3 Aguardar Liberação de Equipe

Se não houver pessoal disponível para trabalhar sobre os requisitos do sistema, será aguardado a disponibilidade de membros das equipes.

4.3.1.4 Definir Funcionalidades do Sistema

A equipe irá coletar as funcionalidades catalogadas durante a fase inicial de Concepção, analisando a documentação produzida e se necessário realizando novas entrevistas com o Demandante, a fim de confirmar as funcionalidades que serão desenvolvidas.

4.3.1.5 Propor MVP

A equipe de desenvolvimento, de posse das principais funcionalidades do sistema, propõe um **Produto Mínimo Possível**, que corresponde a uma primeira entrega, atendendo aos requisitos básicos necessários para que o demandante possa fornecer um feedback e comprovar a real viabilidade do software.

4.3.1.6 Homologar MVP

O Gestor de Desenvolvimento, como condutor do processo de desenvolvimento, e ciente dos requisitos do projeto, terá o papel de homologar e enviar para **aprovação**, pelo Gestor do Sistema, o documento contendo a descrição do MVP.

4.3.1.7 Aprovar MVP

O Gestor do Sistema, de posse do documento, descrevendo as funcionalidades que serão desenvolvidas no MVP, aprova-o ou sugere correções, as quais poderão ser tratadas diretamente com a Equipe e Gestor de Desenvolvimento .

4.3.1.8 Criar cartões e Inserir no Backlog do Quadro

Com base no backlog já produzido na fase de concepção e na definição das funcionalidades verificadas, a equipe irá produzir os cartões correspondentes às funcionalidades que serão desenvolvidas, inserindo-as na área do backlog do quadro kanban.

Ao final, ter-se-á o backlog alimentado com os cartões do sistema.

4.3.2 Realizar Desenvolvimento

É o subprocesso principal, correspondendo a execução de todas as atividades necessárias ao desenvolvimento do Sistema.

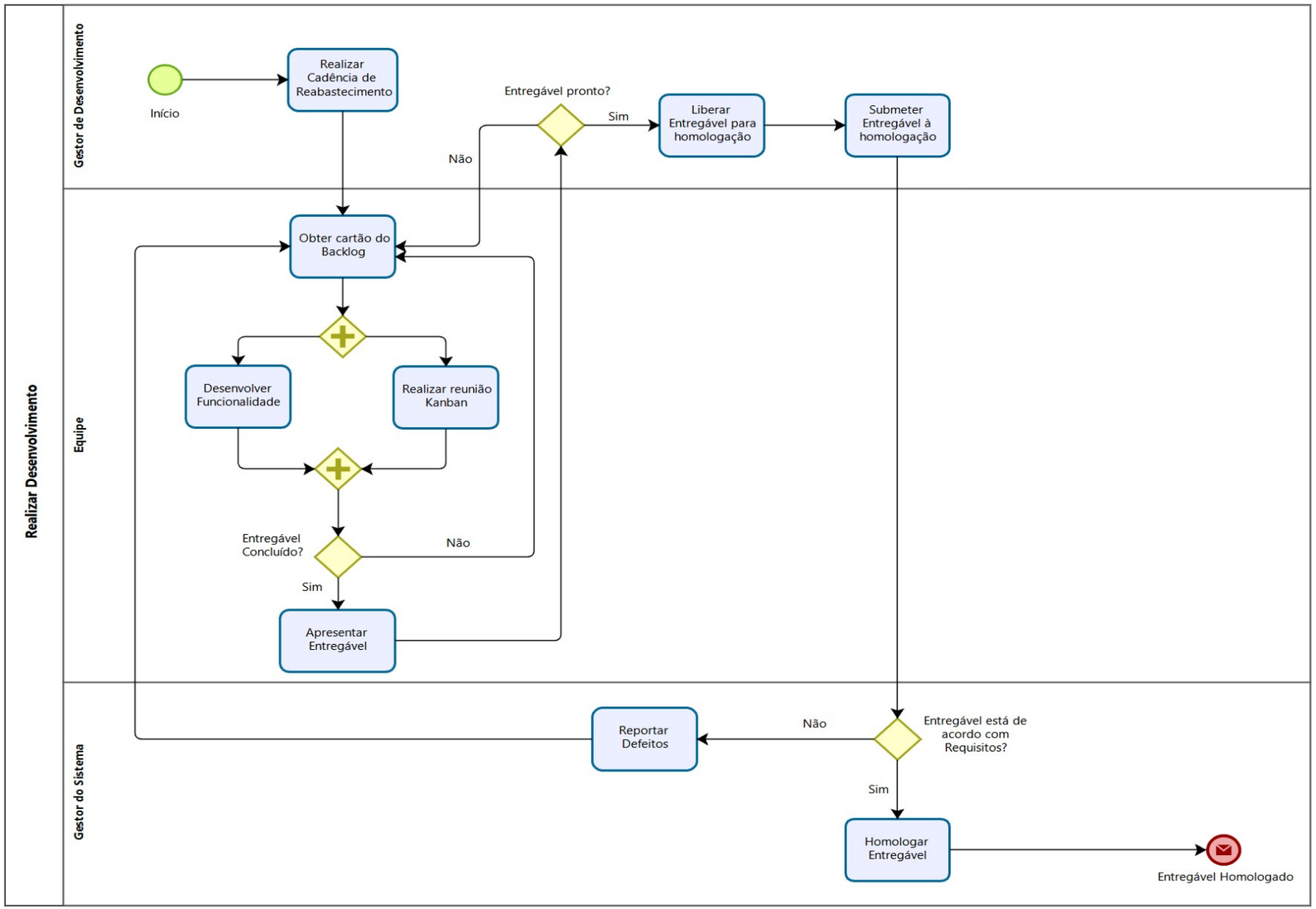


Figura 10 – Fluxo do Subprocesso Realizar Desenvolvimento

Este subprocesso corresponde ao desenvolvimento do sistema em si, por meio de codificação e construção de interfaces.

Neste processo, a equipe de desenvolvimento irá movendo os cartões, no quadro kanban, da coluna inicial, Backlog, até a conclusão da tarefa, quando se dará por finalizada. A quantidade de colunas será definida pela equipe quando da implementação deste Manual.

4.3.2.1 Realizar Cadência de Reabastecimento

O Gestor de Desenvolvimento de Sistema irá reunir a equipe designada e irá alocar todas as atividades necessárias para início ou continuidade do desenvolvimento do sistema, escrevendo os cartões e dispondo na coluna *Backlog* do quadro *kanban*. Será uma atividades cíclica e com periodicidade estabelecida de acordo com a demanda e com duração sugerida de 30 (trinta) minutos.

4.3.2.2 Obter Cartão do Backlog

Concluído o desenvolvimento das funcionalidades de um ou mais cartões e movimentadas para a coluna subsequente, pode-se retirar um novo cartão com uma tarefa a ser executada da coluna de backlog.

A obtenção de novo cartão deverá ser monitorada pelo Gestor de Desenvolvimento, com a nomeação do desenvolvedor designado para execução da tarefa.

4.3.2.3 Desenvolver Funcionalidade

Trata da implementação da funcionalidade para o Sistema a desenvolvido, compreendendo a fase de análise e codificação.

Será uma atividade cíclica e contínua até a construção de uma parte, possível de ser utilizado pelo demandante, ou todo o sistema completo.

4.3.2.4 Realizar Reunião Kanban

Paralela à atividade a execução das tarefas de desenvolvimento de funcionalidades, é realizada, diariamente ou em outro período de tempo predeterminado, reuniões com todas as equipes, objetivando monitorar o andamento das tarefas e solucionar os eventos bloqueadores.

Entende-se por evento bloqueador, algo que impeça ou diminua o tempo de conclusão de uma tarefa.

É boa prática dá enfoque aos eventos bloqueadores e traçar caminhos para solucioná-los.

4.3.2.5 Apresentar Entregável

Caso o conjunto de funcionalidades desenvolvidas constitua em um parte do sistema que já possa ser testada e homologada pelo Gestor do Sistema (Demandante). Essa parte, chamada de Entregável, será submetida à avaliação do Gestor de Desenvolvimento para aprovação.

4.3.2.6 Liberar Entregável para Homologação

Se na avaliação do Gestor de Desenvolvimento, o entregável estiver apto para homologação pelo Demandante (Gestor do Sistema). Em caso negativo, a equipe de desenvolvimento será mobilizada para sanar eventual inconsistência verificada pelo Gestor de Desenvolvimento, voltando para o fluxo inicial, e adicionando novo cartão ao backlog, com a atividade a ser realizada. Essa tarefa terá prioridade sobre todas as outras.

4.3.2.7 Submeter Entregável para Homologação

O Gestor de Desenvolvimento irá submeter o entregável à homologação pelo Demandante (Gestor do Sistema), disponibilizando em ambiente de testes para que possa verificar se todos os requisitos foram atendidos.

4.3.2.8 Reportar Defeitos

Caso o Gestor de Sistemas detecte, durante a homologação, o não atendimento aos requisitos iniciais repassados à equipe de desenvolvimento, ou encontre erros no entregável, será reportado o problema à equipe. Assim, um novo cartão será adicionado ao backlog, tendo prioridade sobre os demais para sua execução.

4.3.2.9 Homologar Entregável

O Demandante (Gestor do Sistema) irá homologar entregável, informando ao Gestor de Desenvolvimento. O subprocesso Entregar Funcionalidade ou Sistema será acionado.

4.3.2.10 Matriz de Responsabilidades

ATIVIDADES/PAPÉIS	GESTOR DE DESENVOLVIMENTO	GESTOR DO SISTEMA	EQUIPE
Realizar Cadência de Reabastecimento	R/E	C	E
Obter Cartão do Backlog	C/I	-	R/E
Desenvolver Funcionalidade	I	-	R/E
Realizar Reunião Kanban	R/E	-	R/E
Apresentar Entregável	I	-	R/E
Liberar para Homologação	R/E	I	I
Submeter à Homologação	R/E	I	I
Reportar Defeitos	I	R/E	C/I
Homologar Entregável	C/I	R/E	I
Legenda: R—Responsável; E—Executor; C—Consultado; I—Informado.			

Tabela 03 – Matriz de responsabilidades do subprocesso Realizar Desenvolvimento

3.12 Recursos e Resultados

ATIVIDADES	RECURSOS	RESULTADOS
Realizar Cadência de Reabastecimento	Documento de História de Usuário	Quadro Kanban alimentado
Obter Cartão do Backlog	Quadro kanban	Cartão movimentado para execução
Desenvolver Funcionalidade	Formulário de Análise Preliminar; Documento De História do Usuário	Tarefas dos cartões codificadas e implementadas
Realizar Reunião Kanban	Quadro Kanban	Próximos cartões escalonados Soluções propostas para eventos bloqueadores Lições aprendidas
Liberar Entregável para Homologação	Quadro Kanban	Entregável disponível para submissão
Submeter Entregável para Homologação	Quadro Kanban	Entregável disponível para homologação
Reportar Defeitos	Entregável disponibilizado	Lista de erros e não atendimento de requisitos fornecidos
Homologar Entregável	Entregável disponibilizado	Entregável homologado

Tabela 04 – Recursos do subprocesso Realizar Desenvolvimento

4.3.3 Entregar Funcionalidade ou Sistema

É o subprocesso executado após o fornecimento de parte do sistema plenamente funcional ou o sistema completo.

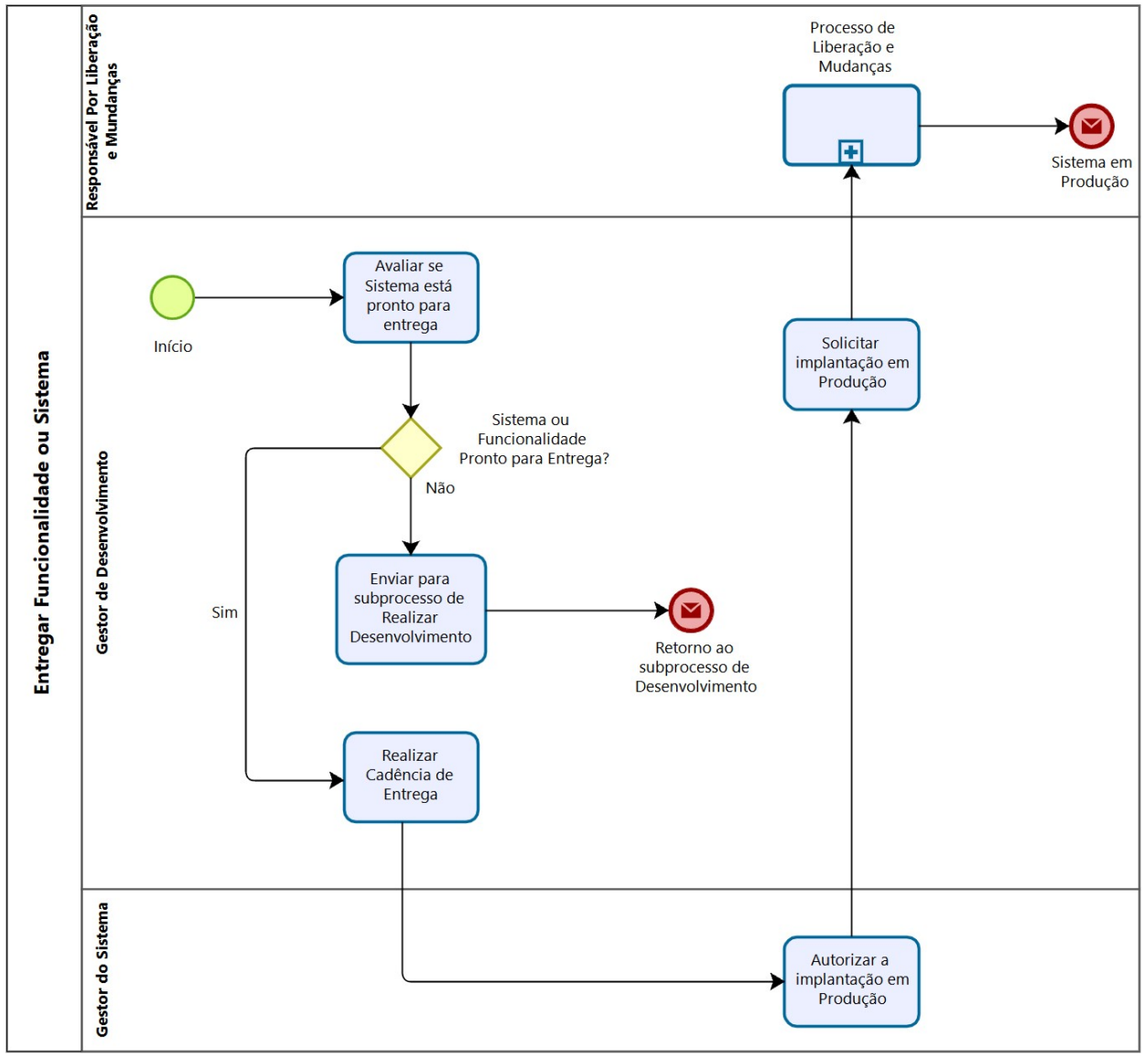


Figura 11 – Fluxo do Subprocesso de Entregar Funcionalidade ou Sistema

4.3.3.1 Avaliar se Funcionalidade ou Sistema está pronto para Entrega

O Gestor de Desenvolvimento realiza última avaliação se a funcionalidade requisitada ou Sistema está pronta para entrega para produção ou deve voltar para o subprocesso de Realizar Desenvolvimento.

4.3.3.2 Enviar para Subprocesso de Realizar Desenvolvimento

Julgado que o sistema ainda não atende aos requisitos ou apresenta defeitos, o Gestor do Sistema mobilizará novamente a equipe de Desenvolvimento para corrigir eventuais problemas verificados no entregável. O subprocesso de Realizar Desenvolvimento será novamente iniciado.

4.3.3.3 Realizar Cadência de Entrega

Nesta etapa, é realizada uma reunião com a equipe de desenvolvimento com o objetivo de efetuar um planejamento para a entrega definitiva do entregável. Deve-se verificar os seguintes pontos a serem observados:

- Quais itens do sistema estão (ou estarão) prontos para serem lançados?
- O que é necessário para realmente liberar cada item em produção?
- Quais testes serão necessários após a liberação para validar a integridade dos sistemas em produção?
- Quais riscos estão envolvidos?
- Como esses riscos estão sendo mitigados?
- Quais planos de contingência são necessários?
- Quem precisa estar envolvido no lançamento e presente durante o deploy em produção?
- Quanto tempo vai demorar a release?

4.3.3.4 Autorizar a implantação em produção

O Gestor do Sistema (Demandante), com base na homologação já realizada, poderá autorizar ao Gestor de Desenvolvimento a implantação do entregável em produção.

4.3.3.5 Solicitar a implantação em Produção

O Gestor de Desenvolvimento solicitará ao responsável pelo processo de Liberação e Mudanças a implantação da funcionalidade ou do Sistema.

O Processo de Liberação e Mudanças será iniciado. Este processo não faz parte do escopo deste Manual, porém já está formalizado no âmbito do TRE-PI.

Após esta última etapa, a funcionalidade ou Sistema estará em produção para uso pelo Demandante.

4.3.3.6 Matriz de Responsabilidades

ATIVIDADES/PAPÉIS	GESTOR DE DESENVOLVIMENTO	GESTOR DO SISTEMA	EQUIPE
Avaliar se Sistema está pronto para entrega	R/E	-	-
Enviar para subprocesso de Realizar Desenvolvimento	R/E	C/I	-
Realizar Cadência de Entrega	R/E	I	-
Autorizar a implantação em Produção	R/E	I	-
Solicitar implantação em Produção	R/E	I	I
Legenda: R—Responsável; E—Executor; C—Consultado; I—Informado.			

Tabela 05 – Matriz de responsabilidades do subprocesso Entregar Funcionalidade ou Sistema

3.3.3.7 Recursos e Resultados

ATIVIDADES	RECURSOS	RESULTADOS
Avaliar se Sistema está pronto para entrega	Documento de Requisitos Funcionalidade ou Sistema no ambiente de homologação	Documento de Avaliação de Sistema Funcionalidade ou Sistema
Enviar para subprocesso de Realizar Desenvolvimento	Erros ou defeitos encontrados Não conformidade com novos requisitos	Relatório de erros ou defeitos encontrados Novos cartões para Backlog
Realizar Cadência de Entrega	Funcionalidade ou Sistema	Relatórios de planejamento de implantação
Autorizar a implantação em Produção	Funcionalidade ou Sistema	Termo de autorização para implantação
Solicitar implantação em Produção	Funcionalidade ou Sistema	Requerimento de Liberação e Mudanças

Tabela 06 – Recursos e Resultados do subprocesso Entregar Funcionalidade do Sistema

4.3.4 Finalizar Sistema

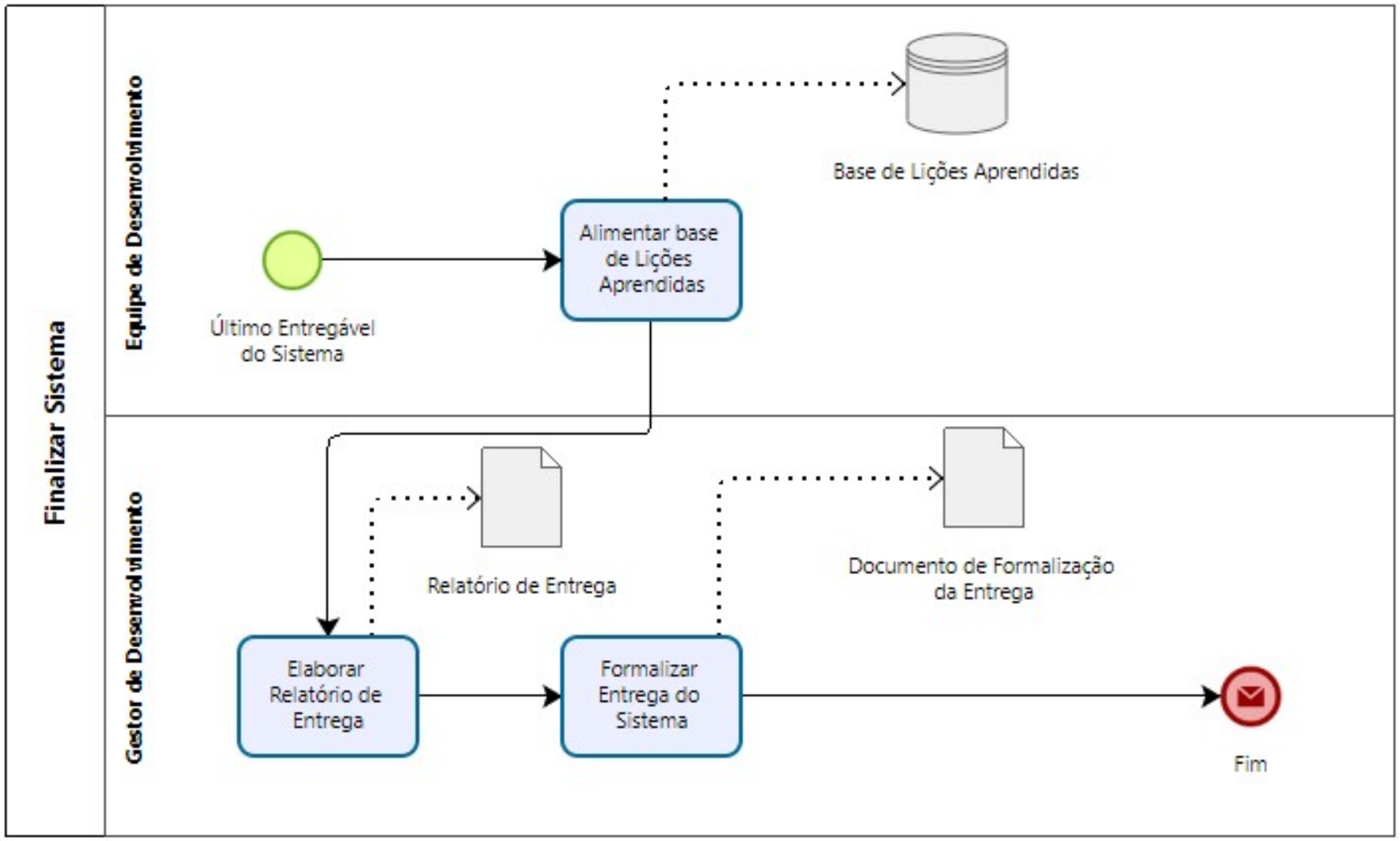


Figura 12 – Fluxo do Subprocesso Encerrar Projeto

4.3.4.1 Alimentar Base de Lições Aprendidas

Corresponde à realização de documentação e registro, pela Equipe de Desenvolvimento, diante de todos os eventos bloqueadores e problemas verificados, as soluções encontradas para resolução de tais obstáculos.

Estes registros poderão feitas em sistema próprio de base de conhecimentos.

4.3.4.2 Elaborar Relatório de Entrega

O Gestor de Desenvolvimento, antes de formalizar a entrega do sistema, elaborará relatório contendo todos o recursos utilizados para desenvolver, tais como: tempo de desenvolvimento, quantidade de desenvolvedores (terceirizados e efetivos) que participaram do desenvolvimento, o custo por hora de trabalho (se terceirizados, a base será o salário previsto em contrato, se servidor efetivo, a base será o valor de sua remuneração constante de lei). Se houve necessidade de aquisição de licenças de software ou ferramentas de desenvolvimento, acrescentar, o valor despendido.

4.3.4.3 Formalizar Entrega do Sistema

Representa a formalização da entrega do produto final junto ao gestor do sistema. O projeto é considerado encerrado quando o último entregável ou o Sistema completo for homologada pelo gestor do sistema e não haja mais pendências. Será preenchido um documento de homologação do sistema pelo gestor do sistema para formalizar o encerramento do projeto.

4.3.4.4 Matriz de Responsabilidades

ATIVIDADES/PAPÉIS	GESTOR DE DESENVOLVIMENTO	EQUIPE
Alimentar Base de Lições Aprendidas	C/I	R/E
Elaborar Relatório de Entrega	R/E	-
Formalizar Entrega do Projeto	R/E	-
Legenda: R—Responsável; E—Executor; C—Consultado; I—Informado.		

Tabela 07 – Matriz de Responsabilidades do Subprocesso de Finalizar Projeto

4.3.4.5 Recursos e Resultados

ATIVIDADES	RECURSOS	RESULTADOS
Alimentar Base de Lições Aprendidas	Problemas e eventos bloqueadores encontrados	Base de Conhecimento alimentada
Formalizar Entrega do Projeto	Funcionalidade ou Sistema	Documento de Formalização de Entrega

Tabela 08 – Recursos e Resultados do Subprocesso de Finalizar Projeto

5. Sustentação do Sistema

5.1 Introdução

A fase de sustentação do sistema consiste na etapa em que o mesmo, uma vez desenvolvido e implantado em ambiente de produção, começa a operar e a cumprir seu propósito. Essa é uma das fases mais longas de seu ciclo de vida, momento em que far-se-ão necessárias manutenções corretivas e/ou evolutivas que visarão, respectivamente, corrigir problemas identificados com o uso ou inserir novas funcionalidades percebidas como necessárias.

O diagrama abaixo ilustra este processo:

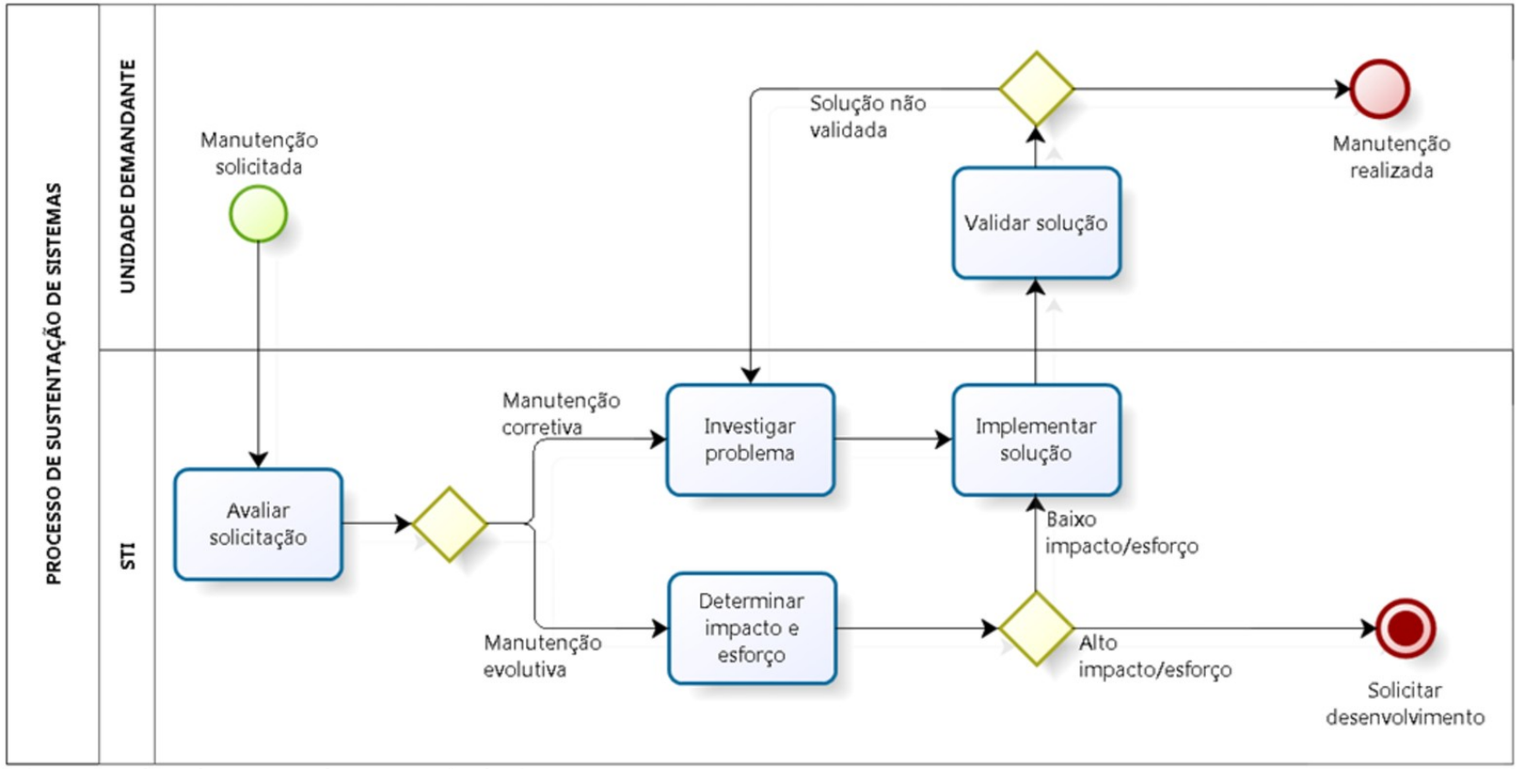


Figura 13 – Fluxo do Processo de Sustentação de Sistemas

5.2 Avaliar Solicitação

A avaliação da solicitação consiste na análise da demanda de manutenção recebida com o objetivo de classificá-la em manutenção 1 – corretiva; ou 2 – evolutiva. A Unidade de Desenvolvimento deverá identificar o tipo de manutenção solicitada e prosseguir com este fluxo de acordo com essa classificação. Esta atividade, uma vez concluída, desencadeará uma das duas atividades abaixo: Investigar Problema – Fluxo ativado caso a solicitação seja classificada como manutenção corretiva; Determinar Impacto/Esforço – Fluxo ativado caso a solicitação seja classificada como manutenção evolutiva.

5.3 Investigar Problema

Uma vez que a solicitação de manutenção foi classificada como corretiva, esta atividade consiste em diagnosticar o problema relatado pela unidade demandante a fim de identificar a solução apropriada.

5.4 Determinar Impacto/Esforço

Uma vez que a solicitação de manutenção foi classificada como evolutiva, faz-se necessária a análise do impacto que essa evolução causará na arquitetura geral do sistema, bem como do esforço necessário para implementá-la. Toda manutenção evolutiva terá, ao final desta tarefa, uma das duas classificações a seguir: Baixo Impacto/Esforço – desencadeará a atividade Implementar Solução; Alto Impacto/Esforço – encerrará este processo e iniciará a atividade ‘Solicitar desenvolvimento de Sistema’, do processo de desenvolvimento de sistemas, tratado no capítulo 4.

5.5 Implementar Solução

Esta atividade é responsável por implementar a solução do problema, definida na atividade ‘Investigar Problema’, ou por desenvolver a nova funcionalidade classificada como de baixo impacto/esforço pela atividade ‘Determinar Impacto/Esforço’. Quando de seu término, esta atividade deve notificar a unidade demandante que uma solução para sua solicitação foi implementada e está sujeita a validação.

5.6 Validar Solução

Esta atividade é responsável por obter da unidade demandante o aceite da solução implementada pela STI. A saída desta atividade pode direcionar o fluxo do processo de volta à atividade 'Implementar Solução', caso a resposta dada ao problema não atenda ao esperado, ou encerrar o processo, quando a solução atende à expectativa da unidade demandante.

5.7 Matriz de Responsabilidades

ATIVIDADES/PAPÉIS	GESTOR DE DESENVOLVIMENTO	EQUIPE
Alimentar Base de Lições Aprendi-das	C/I	R/E
Formalizar Entrega do Projeto	R/E	-
Legenda: R—Responsável; E—Executor; C—Consultado; I—Informado.		

Tabela 09 – Matriz de Responsabilidades do Processo Sustentação do Sistema

5.8 Recursos e Resultados

ATIVIDADES	RECURSOS	RESULTADOS
Alimentar Base de Lições Aprendidas	Problemas e eventos bloqueadores encontrados	Base de Conhecimento alimentada
Formalizar Entrega do Projeto	Funcionalidade ou Sistema	Documento de Formalização de Entrega

Tabela 10 – Recursos e Resultados do Processo Sustentação do Sistema

6. Declínio do Sistema

6.1 Introdução

Considera-se que um sistema entra em declínio quando é preciso atender novas necessidades advindas de evoluções tecnológicas, organizacionais ou legais. Há situações em que se torna muito oneroso realizar adaptações nos sistemas para atender tais requisitos. Pode ocorrer também do sistema não ser mais necessário para o cliente, devendo apenas ser retirado do ambiente de produção. A decisão por desativar um sistema pode partir do próprio gestor do sistema (ou gestor de negócio) ou então o Gestor Técnico do Sistema. A seguir, são descritas as etapas que serão executadas quando um sistema já implantado entra em declínio.

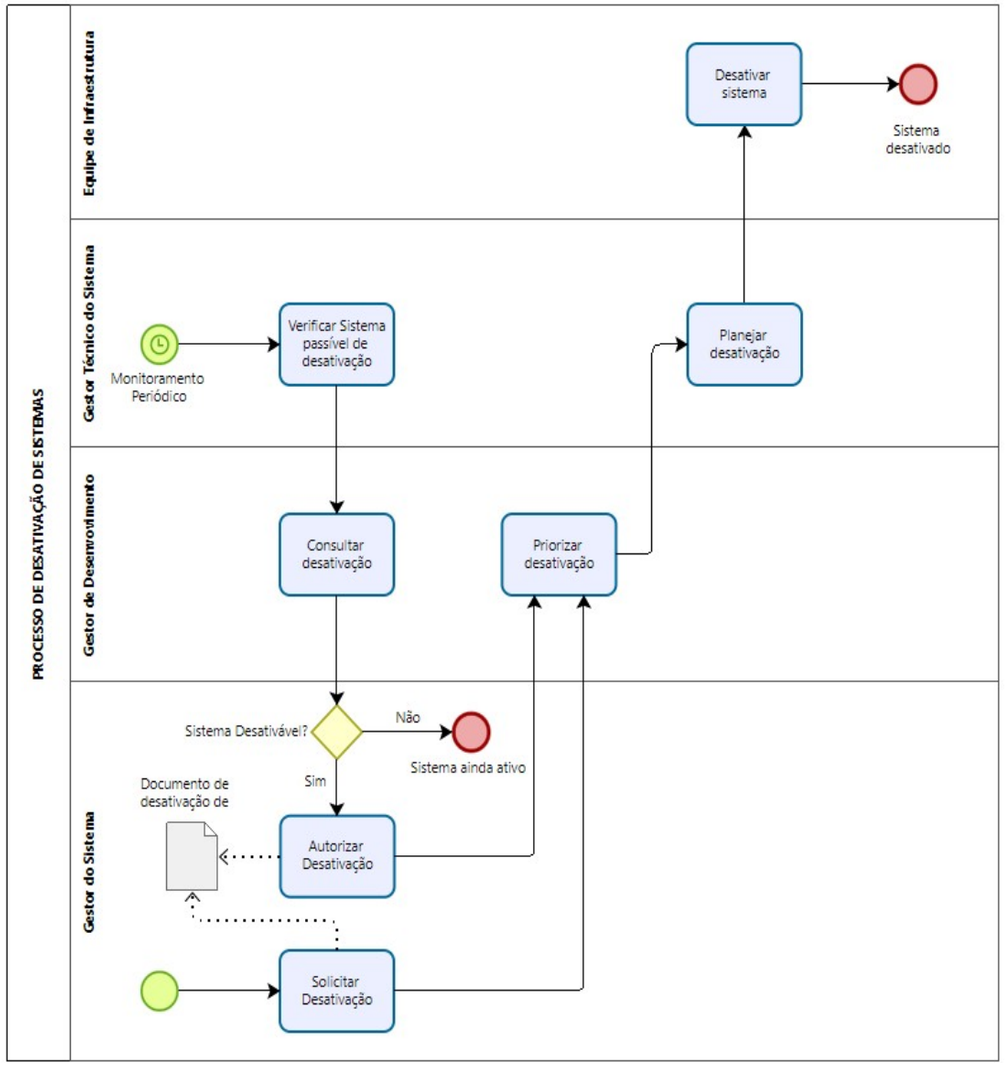


Figura 14 – Fluxo do Processo de Desativação de Sistemas

6.2 Verificar Sistema Passível de Desativação

O Gestor Técnico irá, periodicamente, verificar quais sistemas podem ser desativadas, por falta de uso ou obsolescência, entrando em contato com o Gestor de Desenvolvimento sobre a possibilidade de desativação.

6.3 Consultar Desativação

O Gestor técnico comunicará ao Gestor de Desenvolvimento qual sistema poderá ser desativado, e este consultará ao Gestor do Sistema se realmente o sistema poderá ser desativado.

6.4 Autorizar Desativação

O Gestor técnico comunicará ao Gestor de Desenvolvimento qual sistema poderá ser desativado, e este consultará ao Gestor do Sistema se realmente o sistema poderá ser desativado.

Se o Gestor do Sistema autorizar a desativação, será comunicado ao Gestor de Desenvolvimento, passando para o passo seguinte.

Caso não seja autorizado, o processo será finalizado, sem a desativação do sistema.

6.5 Solicitar Desativação

Atividade em que a Unidade Demandante, na figura do Gestor do Sistema, formalmente se manifesta solicitando a desativação de um sistema sob sua responsabilidade. Essa formalização se dá através do preenchimento e entrega, via sistema de processo administrativo, do Documento de Desativação de sistemas.

6.6 Priorizar Desativação

Uma vez formalizado junto ao Gestor de Desenvolvimento, o pedido para desativação de um sistema, é necessário que o gestor da referida unidade verifique as demandas correntes para que a atividade seja priorizada e encaminhada ao gestor técnico do sistema para definir o planejamen-

6.7 Planeja a Desativação

Nesta etapa, o gestor técnico analisa o impacto da desativação do sistema e elabora um cronograma para que a transição ocorra de maneira segura, sem transtornos para o usuário e respectivas unidades impactadas. O planejamento para a desativação do sistema deve contemplar: As atividades a serem executadas e respectivos responsáveis; Prazos para execução; Se necessário, definição de uma etapa piloto a fim de analisar possíveis impactos não previstos inicialmente; Política de backup dos dados.

6.8 Desativar Sistema

Corresponde à execução do planejamento definido na etapa anterior no que diz respeito ao backup dos dados e liberação de recursos dos servidores de produção. Será realizada pela unidade de infraestrutura do TRE-PI.

6.9 Matriz de Responsabilidade

ATIVIDADES/PAPÉIS	GESTOR DO SISTEMA	GESTOR DE DESENVOLVIMENTO	GESTOR TÉCNICO	EQUIPE DE INFRAESTRUTURA
Verificar Sistema Passível de Desativação	-	I	R/E	-
Consultar Desativação	-	C	R/E	-
Autorizar Desativação	R/E			
Solicitar desativação	R/E	C/I	C/I	-
Priorizar desativação	C/I	R/E	C/I	-
Planejar desativação	C	C/I	R/E	I
Desativar sistema	I	I	C/I	R/E

Tabela 11 – Matriz de responsabilidades do processo de Declínio de um sistema

6.10 Recursos e Resultados

ATIVIDADES	RECURSOS	RESULTADOS
Verificar Sistema Passível de Desativação	Não atualização de dados no sistema Relatório de acessos ao sistema	Relatório de sistemas passíveis de desativar
Consultar Desativação	Relatório de sistemas passíveis de desativar	Documento enviado ao Gestor de Sistema consultando sobre desativação
Autorizar Desativação	Documento enviado ao Gestor de Sistema consultando sobre desativação	Documento de Desativação
Solicitar desativação de sistema	Justificativas para a necessidade de desativação do sistema.	Documento de solicitação de desativação de sistema;
Priorizar desativação	Documento de solicitação de desativação de sistema; PADS atualizado; Relação de outras demandas do setor.	Desativação priorizada entre as outras demandas da unidade.
Planejar desativação	Documento de solicitação de desativação de sistema; Desativação priorizada entre as outras demandas da unidade.	Cronograma de desativação do sistema.
Desativar sistema	Cronograma de desativação do sistema.	Sistema desativado.

Tabela 12 – Recursos e Resultados do processo de Declínio de um sistema

7. Escopo e Requisitos do Sistema

7.1 Introdução

Fundamentais ao completo entendimento da necessidade do cliente, o escopo e os requisitos configuram dois dos principais artefatos que precisam ser bem definidos e gerenciados. Isso posto, e considerando ainda as boas práticas de desenvolvimento de software, faz-se necessária a efetiva gestão dos mesmos. Essa gestão busca estabelecer um entendimento comum da natureza do sistema, seus limites e funcionalidades, bem como manter esse entendimento atualizado conforme as mudança ocorrem. Busca-se também, dessa forma, manter o controle das futuras evoluções que o mesmo venha a receber.

7.2 Gerenciamento de Escopo e Requisitos

O gerenciamento do escopo tem por objetivo estabelecer de maneira clara para as partes interessadas os limites do sistema. Isso significa que ambos, cliente e unidade de desenvolvimento, tenham uma visão consensual das capacidades esperadas para o sistema.

É importante que, quando da definição do escopo, fique claro o que não deve fazer parte do escopo do sistema. Isso reduz ambiguidades e aumenta a clareza, alinhando as expectativas. A definição do escopo acontece quando da atividade de definição do documento de visão e arquitetura, no processo de desenvolvimento de sistemas. O escopo pode ainda ser ampliado ou reduzido com eventuais manutenções evolutivas. A depender do tipo e abrangência de determinada evolução, essa pode alterar significativamente as fronteiras do sistema, adicionando ou removendo funcionalidades relevantes. Essas mudanças devem, necessariamente, ser refletidas no escopo documentado.
Ao passo que o escopo define os limites do sistema, os requisitos estabelecem as funcionalidades que os usuários esperam que o mesmo executem (requisitos funcionais) e as restrições sob as quais o mesmo tenha que operar (requisitos não funcionais). O gerenciamento desses requisitos envolve identificar, controlar e acompanhar suas necessidades e mudanças.

Essa gestão, semelhante à do escopo, ocorre de maneira dispersa em todo o ciclo de vida do produto. Os requisitos são inicialmente levantados na fase de concepção do projeto e documentados no artefato denominado documento de análise preliminar. Esses requisitos são posteriormente especificados de maneira mais detalhada, considerando as restrições impostas pela arquitetura do sistema e por outras definições que irão compor o documento de visão e arquitetura do sistema. Por fim, existe a constante alteração de requisitos, cenário comum em projetos de software. Sejam alterações de requisitos que surgem ao longo do desenvolvimento do software, sejam necessidades de mudanças após este estar em produção, ambas precisam ser gerenciadas de maneira coordenada. Todo o exposto deixa claro que a gestão de escopo e requisitos ocorre de maneira fragmentada ao longo do ciclo de vida do produto, tendo suas atividades detalhadas nos capítulos anteriores, mas pode ser ilustrada no diagrama abaixo:

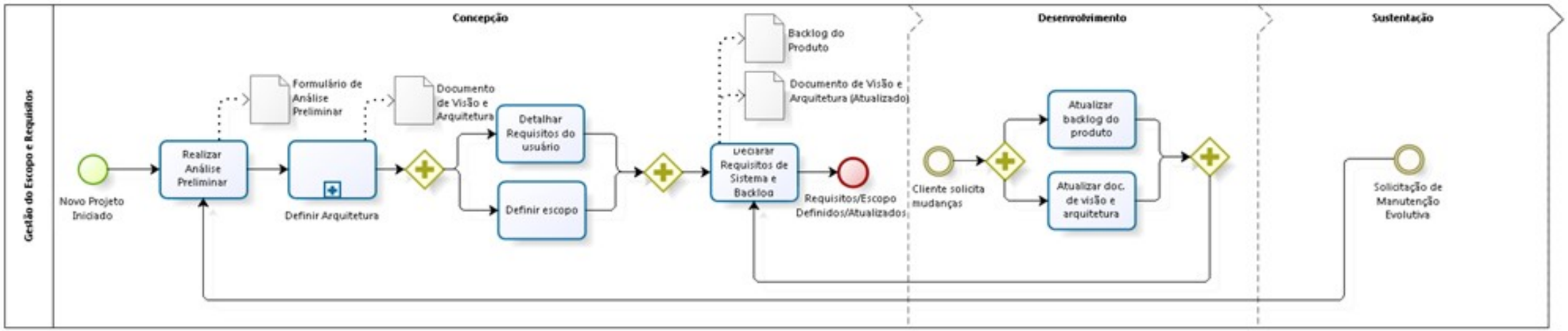


Figura 15 – Fluxo do Processo de Gerenciamento de Escopo

8. Diretrizes para o Desenvolvimento de Software

8.1 Introdução

A busca pela conformidade, no âmbito de um produto de software, tem por objetivo garantir que o mesmo atende aos requisitos estabelecidos em normas, procedimentos e na legislação aplicável. Em síntese, a conformidade é a avaliação da adequação do software aos requisitos de legislação e outros tipos de padronização aplicáveis. Nesse contexto, esta sessão estabelece as diretrizes que devem nortear as atividades de desenvolvimento de software, com o objetivo de garantir essa conformidade.

8.2 Diretrizes de Conformidade

Esta sessão define, de maneira não exaustiva, as diretrizes a serem observadas ao desenvolver soluções de software. Estimular a adoção de metodologia de desenvolvimento de sistemas, buscando assegurar padronização, integridade e segurança; Criar e manter, para toda aplicação em produção, documentação que oriente o usuário no uso do sistema; Adotar padrões abertos no desenvolvimento de tecnologia da informação; Observar as recomendações da Política de Segurança da Informação instituído pelo TRE-PI;

Garantir que as aplicações desenvolvidas sejam responsivas; Garantir que as aplicações desenvolvidas sejam adequadas à exibição em dispositivos móveis; Observar as recomendações do Modelo de Acessibilidade em Governo Eletrônico (e-MAG); Observar as recomendações do Modelo de Requisitos para Sistemas Informatizados de Gestão de Processos e Documentos da Justiça Federal – MoReq-Jus; Observar o Modelo Nacional de Interoperabilidade, de acordo com as metas do termo de cooperação técnica nº 58/2009; Sempre que aplicável, prover suporte à assinatura digital, conforme padrões estabelecidos pela ICP Brasil.

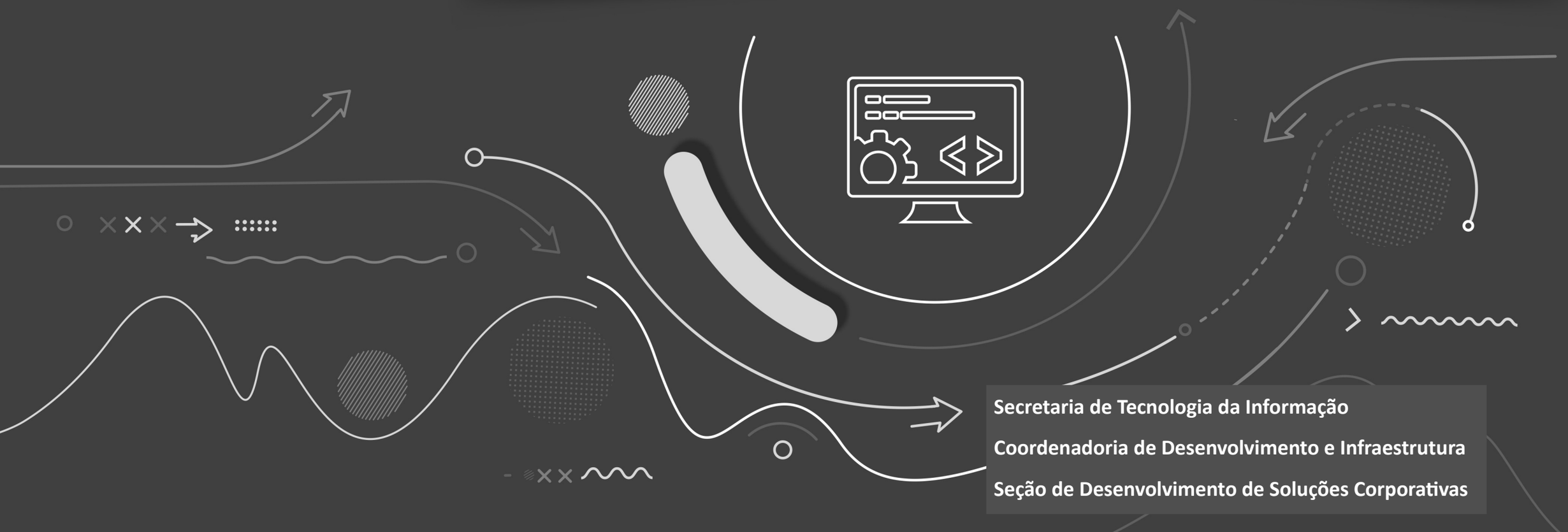
9. Conclusão

Foram estabelecidos neste Manual do Processo de Software (MPS) os principais padrões que norteiam a trajetória do desenvolvimento de uma solução corporativa, desde a formalização da solicitação da demanda; ingresso na lista priorizada pelo CDTI; desenvolvimento; gestão e, por fim, sua descontinuidade. Importante ressaltar que a metodologia aqui estabelecida deve evoluir conforme mudanças normativas, de boas práticas ou ainda na rotina de trabalho deste Regional ocorram. É importante que este padrão reflita sempre as melhores práticas de desenvolvimento de software, bem como represente a realidade vivida pelos profissionais que dele fazem uso. Na versão inicial deste documento, algumas técnicas previstas em métodos ágeis foram incorporadas ao processo já existente, buscando melhorar o trabalho desenvolvido pela unidade. Num segundo momento, foi introduzido o padrão de arquitetura de software e também definido o conceito de ciclo de vida do sistema. A presente atualização formaliza a evolução de uma metodologia de desenvolvimento para um processo de software, em que o ciclo de vida é todo mapeado na forma de processo e as fases do sistema são melhor apresentadas.

Por estar em constante busca por uma maior maturidade, a adoção desta Metodologia representa um desafio na atuação da STI e no aperfeiçoamento do trabalho de seus integrantes, demandando especial atenção na sua implementação. Portanto, a metodologia aqui descrita deve ser uma ferramenta de trabalho utilizada, na prática, por todos os envolvidos nos projetos de desenvolvimento de software deste Regional, bem como deve ser revista e atualizada periodicamente, baseando-se nas lições aprendidas e na melhoria do processo, buscando alinhar seu conteúdo ao contexto de atuação do Tribunal e das especificidades da STI.

TRE-PI | STI Secretaria da Tecnologia da Informação

MANUAL DO PROCESSO DE DESENVOLVIMENTO DE SOFTWARE



Secretaria de Tecnologia da Informação

Coordenadoria de Desenvolvimento e Infraestrutura

Seção de Desenvolvimento de Soluções Corporativas