SCTi
Superintendência Central
de Tecnologia da Informação

SEGPLAN
SECRETARIA DE ESTADO DE
GESTÃO E PLANEJAMENTO

Processo de Desenvolvimento de Software

Documento de Execução

Gerência de Projetos e Sistemas
Superintendência Central de Tecnologia da Informação
Secretaria de Estado de Gestão e Planejamento

Versão	Data	Alteração no documento	Autor
0.1.0	01/03/2016	Criação do documento.	Paulo Henrique Borges de Melo
0.5.0	01/06/2016	Melhorias e revisões diversas	Paulo Henrique Borges de Melo
0.6.0	01/09/2016	Melhorias e revisões diversas	Paulo Henrique Borges de Melo
1.0.0	01/10/2016	Primeira versão estável do processo	Paulo Henrique Borges de Melo
1.1.0	19/02/2018	Readaptação do processo de deploy	Paulo Henrique Borges de Melo
1.2.0	27/09/2018	Melhoria no Texto e nas Imagens	Gilmar Ferreira Arantes
1.3.0	15/02/2019	Alterações para adaptação à nova estrutura da GPS.	Gilmar Ferreira Arantes

Índice

1 Introdução

O Desenvolvimento de Software é uma atividade de natureza complexa pois envolve o atendimento às necessidades dos Clientes, necessitando da execução de um conjunto considerado de atividades, na maioria das vezes complexas. O entendimento desta complexidade envolve a participação de profissionais com diversas especialidades. Envolve ainda a administração da qualidade estabelecida pelos Clientes, em contraste à disponibilidade de prazos, recursos e orçamento cada vez mais restritos e escassos.

Para lidar com estes interesses e forças muitas vezes conflitantes, no sentido de evitar ou minimizar os impactos de problemas que naturalmente surgem desta atividade, surge a necessidade de se disciplinar o desenvolvimento de software, de tal forma a torná-lo gerenciável. A melhor forma de fazê-lo é a definição de um processo para o desenvolvimento.

Este documento apresenta o PDMS – Processo de Desenvolvimento e Manutenção de Software da Gerência de Projetos e Sistemas (GPS), da Superintendência Central de Tecnologia da Informação (SCTI), da Secretaria de Gestão e Planejamento (SEGPLAN), do Estado de Goiás.

O PDMS foi elaborada com base na Metodologia Ágil *Scrum*. Dentro das possibilidades da GPS, as características desta Metodologia são mantidas no PDMS, como por exemplo:

- Projetos desenvolvidos por equipes multidisciplinares;
- Participação efetiva do Cliente;
- Entregas planejadas através de *Sprints*.

O objetivo principal que motivou a definição e implantação deste processo foi a incansável busca pela qualidade, pela profissionalização da atividade de desenvolvimento de software, no âmbito do poder público estadual. A médio e longo prazo são esperados resultados que incluem: redução de custos, cumprimento de prazos, satisfação dos usuários e principalmente melhoria nos serviços prestados aos cidadãos, através da utilização dos softwares desenvolvidos pela GPS.

Este processo é o resultado do esforço de vários profissionais envolvidos na definição e mapeamento dos processos, na definição de artefatos e padrões. Tudo sob a coordenação do então Gerente Paulo Henrique Borges Melo.

A apresentação visual dos fluxos componentes do processo foi elaborada utilizando a notação BPM, através da ferramenta de modelagem Bizagi. É importante observar que nestes diagramas não existe a separação de cada papel em uma raia específica. Esta decisão foi tomada exclusivamente por questões visuais, para facilitar a utilização da imagem, nesta documentação do processo. De outra forma, os diagramas ficariam muito extensões, o que dificultaria a geração e utilização das suas imagens neste documento.

Com a finalidade facilitar a leitura e o entendimento do processo, este Documento, foi estruturado da Seguinte Forma:

1. Nesta Seção 1 é introduzido o processo, propriamente dito.
2. A Seção 2 – Apresenta uma visão macro do processo, através do Diagrama 1.
3. A Seção 3 – Apresenta o fluxo de solicitações e demandas para um novo projeto, conforme pode ser visualizado no Diagrama 2.
4. A Seção 4 – Apresenta o Fluxo de Solicitações e Demandas para Melhorias, Consultas e Novas Funcionalidades, conforme pode ser observado no Diagrama 3.
5. A Seção 5 – Apresenta o Fluxo de Solicitações e Demandas para Manutenções Corretivas, conforme pode ser observado no Diagrama 4.
6. A Seção 6 – Apresenta o Fluxo de Desenvolvimento, conforme pode ser observado no Diagrama 5.
7. A Seção 7 – Apresenta o Fluxo de Execução de Sentenças SQL para Consultas, Alterações, Exclusões ou Inclusões, conforme pode ser observado no Diagrama 6.
8. A Seção 8 – Apresenta o Fluxo de *Deploy*, conforme pode ser observado no Diagrama 7.
9. Finalmente na Seção 9, contem os anexos a este documento.

2 Processo de Desenvolvimento

O PDMS é composto por seis fluxos, conforme pode-se observar no Diagrama 1. Cada um destes fluxos está detalhado em uma Seção própria, conforme descrito na Introdução, deste Documento.

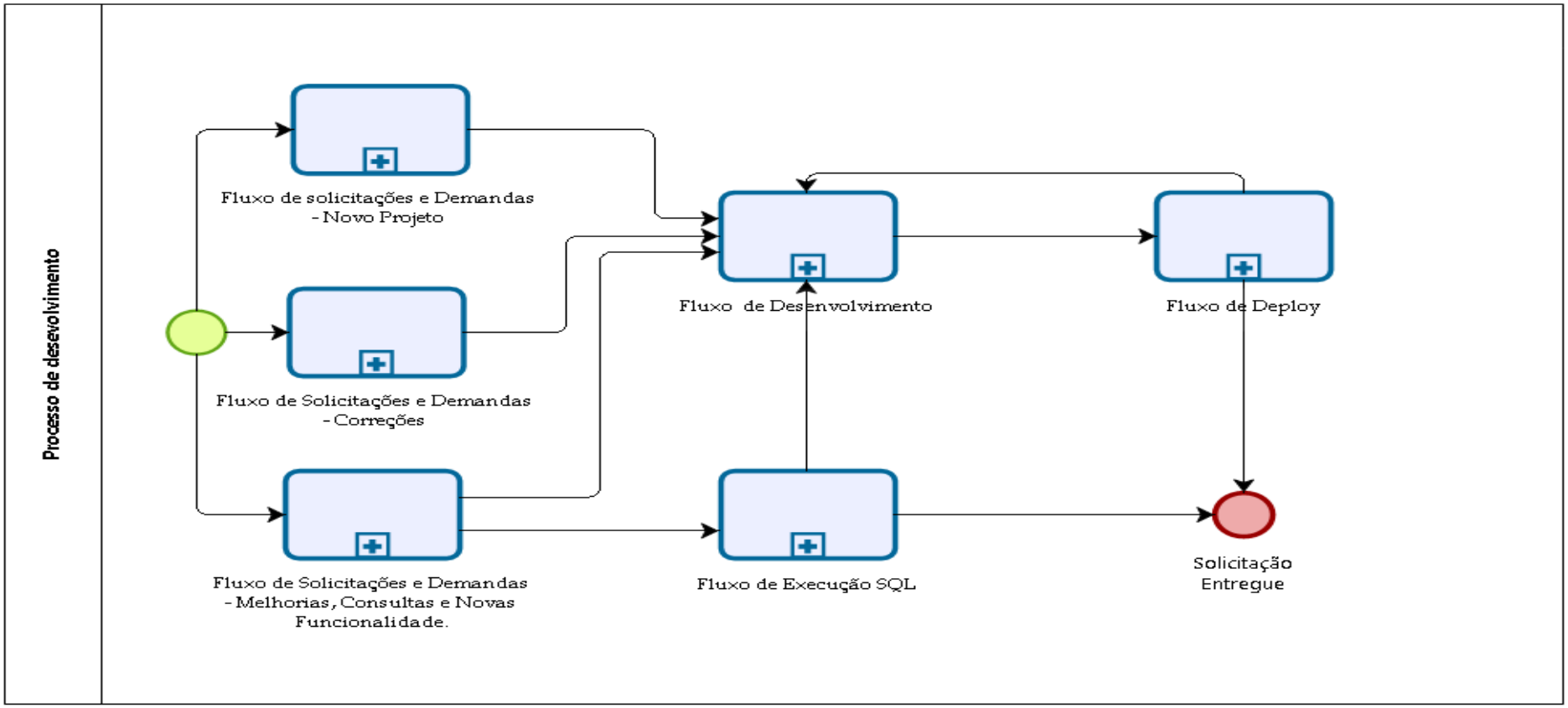

Diagrama 1 – Visão Geral do Processo de Desenvolvimento

3 Fluxo de Solicitações e Demandas – Novo Projeto

O PDMS possui um Fluxo de Solicitações e Demandas para um novo projeto, que é apresentado visualmente pelo Diagrama 2. Este Fluxo contém uma série de atividades e tarefas que estão descritas a partir da Subseção 3.1.

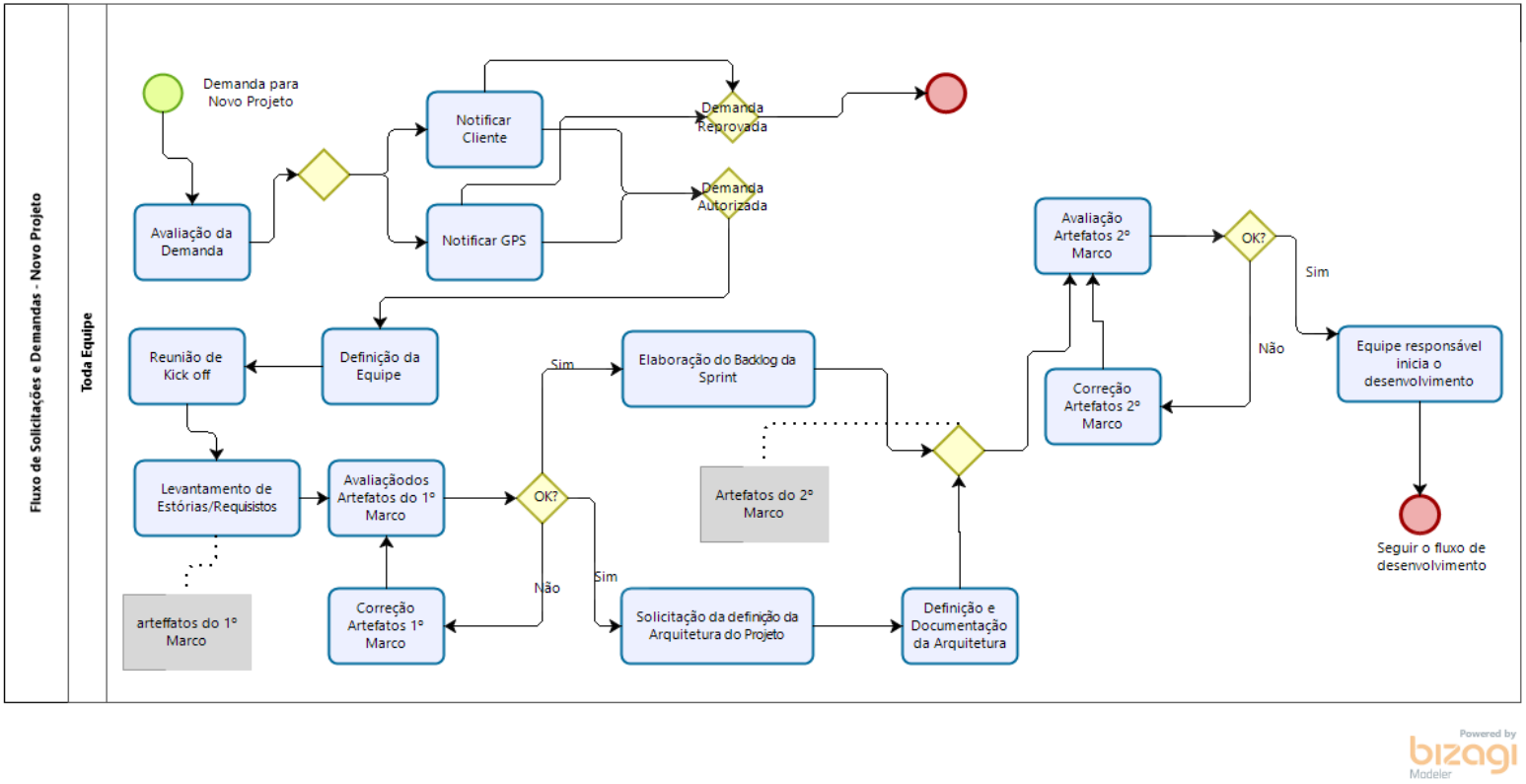

Diagrama 2 - Fluxo de Solicitações e Demandas para um Novo Projeto

3.1 Apresentação da Demanda

3.1.1 O Que:

Este fluxo inicia-se sempre a partir da apresentação de uma demanda por um novo projeto.

3.1.2 Quem:

Esta demanda é apresenta pelo Cliente da SCTI, que são todos os órgãos e servidores da Administração Pública do Estado de Goiás.

3.1.3 Quando:

Sempre que o Cliente detectar uma necessidade.

3.1.4 Como:

A demanda é apresentada através de um Ofício protocolado no Sistema Eletrônico de Informações – SEI.

3.1.5 Entradas:

Processo no SEI apresentando a demanda.

3.1.6 Saídas:

Esta atividade não possui nenhuma saída, por ser a inicial.

3.1.7 Fluxo:

Seguir para a Atividade 3.2.

3.2 Avaliação da Demanda

3.2.1 O Que:

Atividade de Avaliação da Demanda apresentada pelo Cliente.

3.2.2 Quem:

Esta avaliação é feita pelo Superintendente subsidiado por informações obtidas junto ao Gerente de Projetos e Sistemas.

3.2.3 Quando:

Quando existirem demandas para serem avaliadas.

3.2.4 Como:

O Superintendente avalia a demanda, levando em consideração a pertinência da mesma, a possibilidade ou não de atendê-la. Esta avaliação conta com o suporte do Gerente de Projetos e Sistemas.

3.2.5 Entradas:

Demanda a ser avaliada.

3.2.6 Saídas:

Resultado da avaliação, que pode ser aprovação ou reprovação. Neste momento, este resultado é apenas a decisão do Superintendente, não sendo materializada em nenhum artefato específico.

3.2.7 Fluxo:

Seguir para Atividade 3.3 e para Atividade 3.4, pois são executadas em paralelo.

3.3 Notificar o Cliente

3.3.1 O que:

Notificar o Cliente sobre o resultado da avaliação da demanda.

3.3.2 Quem:

Superintendente.

3.3.3 Quando:

Após a avaliação da demanda recebida.

3.3.4 Como:

Através de resposta elaborada ao processo no SEI, informando que a demanda foi ou não aprovada, acompanhada das motivações e justificativas necessárias.

3.3.5 Entradas:

Saídas da Atividade 3.2, descritas na Subseção 3.2.6.

3.3.6 Saídas:

A saída desta atividade está descrita na Tabela 3.1.

Tabela 3.1: Artefatos produzidos pela a Atividade 3.3.

Artefato	Quem?	Prazo
Ofício de resposta ao Cliente no processo do SEI.	Superintendente	Indeterminado

3.3.7 Fluxo:

Se a demanda foi aprovada, o fluxo segue para a Atividade 3.5, Se não, o fluxo é encerrado.

3.4 Notificar o Gerente de Projetos e Sistemas

3.4.1 O Que:

Notificar o Gerente de Projetos e Sistemas sobre o resultado da avaliação da demanda.

3.4.2 Quem:

Superintendente.

3.4.3 Quando:

Após a avaliação da demanda recebida.

3.4.4 Como:

O Gerente de Projetos e Sistemas é notificado através de e-mail, referenciando o processo no SEI.

3.4.5 Entradas:

Saídas da Atividade 3.2, descritas na Subseção 3.2.6.

3.4.6 Saídas:

A saída desta atividade está descrita na Tabela 3.2.

Tabela 3.2: Artefatos produzidos pela Atividade 3.4.

Artefato	Quem?	Prazo
Mensagem de e-mail notificando o GPS.	Superintendente	Indeterminado

3.4.7 Fluxo:

Se a demanda foi aprovada, o fluxo segue para a Atividade 3.5, Se não, o fluxo é encerrado.

3.5 Definição da Equipe

3.5.1 O Que:

Definição da Equipe que será responsável pelo desenvolvimento do projeto.

3.5.2 Quem:

Os envolvidos nesta atividade são:

1. Gerente de Projetos e Sistemas.
2. Líder da Equipe do Projeto.
3. Líder da Equipe da Arquitetura e Qualidade.

3.5.3 Quando:

Imediatamente após a recepção da notificação da aprovação da demanda.

3.5.4 Como:

Esta definição é feita através de reunião do Gerente de Projetos e Sistemas com os envolvidos, descritos na Subseção 3.5.2.[1]

1 Neste passo também poderá ser definida a tecnologia a ser utilizada no projeto.

3.5.5 Entradas:

Saídas da Atividade 3.4, descritas na Subseção 3.4.6.

3.5.6 Saídas:

As saídas geradas a partir da execução desta atividade, estão descritas na Tabela 3.3.

Tabela 3.3: Tarefas relativas a Atividade 3.5.

Tarefa	Quem?	Prazo
Definição da equipe	Todos os participantes	Até o final desta reunião
Envio de e-mail ao Cliente para agendamento de reunião de *Kick off*.	Gerente de Projetos e Sistemas	4 horas

3.5.7 Fluxo:

Seguir para a Atividade 3.6.

3.6 Reunião de *Kick off*

3.6.1 O Que:

Reunião para oficializar o início do projeto, definições iniciais de escopo e cronograma. Agendada pelo Gerente de Projetos e Sistemas[2].

3.6.2 Quem:

Os interessados no projeto, que deverão participar desta reunião são os profissionais listados a seguir:

1. *Product Owner*. (Cliente)
2. Gerente de Projetos e Sistemas.
3. Líder da Equipe a que desenvolverá o projeto.
4. Analista de Sistemas.
5. Analista de Qualidade.
6. Arquiteto de Software.
7. Administrador de Dados.

3.6.3 Quando:

Após a definição da equipe, descrita na Subseção 3.5.

3.6.4 Entradas:

Agendamento de reunião efetuado através da ferramenta de *email*.

2 A primeira reunião para levantamento de requisitos (ver passo 3) pode ser realizada conjuntamente à reunião de Kick off.

3.6.5 Saídas:

Um conjunto de artefatos deve ser gerado como saída desta atividade. Estes artefatos estão identificados na Tabela 3.4.

Tabela 3.4: Artefatos produzidos durante a Atividade 3.6.

Artefato	Quem?	Prazo
Escopo inicial (*Backlog* do Produto).	*Product Owner*	Até o final desta reunião
Cronograma inicial (Planejamento das *Sprints*)	Líder da Equipe	1 dia útil
Prazo (quando já existe uma data fim definida)	*Product Owner*	Até o final desta reunião
Documento de Visão – Ver http://git.intra.goias.gov.br/templates/docs.[3]	Analisa de Sistemas	1 dia útil
Documento *Checklist* de qualidade.	Analista de Qualidade	2 dias úteis

Além dos artefatos gerados, um conjunto de tarefas, também é definido como saída desta atividade. Estas tarefas estão identificadas na Tabela 3.5.

Tabela 3.5: Tarefas definidas na Atividade 3.6.

Tarefa	Quem?	Prazo
Definição de datas e horários de reuniões para levantamento de Estórias/Requisitos (*Sprint Planning Meeting*).	Analista de Sistemas	2 dias úteis
Criar projeto no *Redmine*.	Gerente de Sistemas	1 dia útil
Criar repositórios do projeto no *git* (documentação, projeto de desenvolvimento e projeto de testes)	Arquiteto de Software	1 dia útil
Realizar reunião de alinhamento com a infraestrutura.	Arquiteto de Software	3 dias úteis
Ata de reunião aprovada por todos os participantes.	Líder da Equipe	1 dia útil

3.6.6 Fluxo:

Seguir para a Atividade 3.7.

3.7 Levantamento de Estórias/Requisitos (*Sprint Planning Meeting*)

3.7.1 O Que:

Reuniões para levantamento das Estórias/Requisitos a fim de compor o *Backlog* do Produto e *Backlog* da *Sprint*. Marcadas pelo Líder da Equipe.

3.7.2 Quem:

Os interessados no projeto que deverão participar destas atividades são:

1. *Product Owner*.
2. Analista de Sistemas.

3 Todos os documentos deverão ser armazenados no repositório de documentação do projeto.

3. Líder da Equipe.
4. Gerente de Projetos e Sistemas (Opcional).
5. Analista de Qualidade (Opcional).
6. Administrador de Dados (Opcional).
7. Desenvolvedor (Opcional).

3.7.3 Quando:

Após a reunião de *kick off*.

Essa reunião deverá ser repetida ao final de cada *Sprint* para definição do *Backlog* da próxima *Sprint* (escopo de entrega) e (possível) redefinição do *Backlog* do Produto.

3.7.4 Como:

Interação entre os componentes da equipe e o cliente para o levantamento de estórias/requisitos e o planejamento da ordem de implementação dos mesmos.

3.7.5 Entradas:

Solicitação de agendamento de reunião.

3.7.6 Saídas:

As saídas desta atividade são concretizadas na forma dos artefatos, identificados na Tabela 3.6.

Tabela 3.6: Artefatos gerados durante a Atividade 3.7.

Artefato	Quem?	Prazo
Lista de Estórias/Requisitos que (re)comporão o *Backlog* do Produto.[4]	Analista de Sistemas	1 dia útil
Lista de Estórias/Requisitos que comporão a próxima *Sprint* (*Backlog* da *Sprint*)	Analista de Sistemas e Líder da Equipe	1 dia útil
Ata de cada reunião assinada por todos os participantes contendo os requisitos e regras levantados.[5]	Analista de Sistemas	1 dia útil

3.7.7 Fluxo:

O final desta atividade é estabelecido como o primeiro Marco (*Milestone*) do projeto, quando os artefatos gerados devem ser validados. Desta forma, o fluxo segue para a Atividade 3.8.

4 O *Backlog* do Produto deverá ser cadastrado no *Redmine*.
5 Deverá haver uma tarefa no *Redmine* para cada reunião de requisitos e, após a finalização da reunião, a ata deverá ser digitalizada e anexada à tarefa.

3.8 Avaliação dos Artefatos do 1º Marco

3.8.1 O Que:

Todos os produtos de trabalho (artefatos) gerados até este momento são submetidos ao controle de qualidade.

3.8.2 Quem:

Equipe de Arquitetura e Qualidade.

3.8.3 Quando:

Sempre que um produto de trabalho for disponibilizado, Mas o *deadline* é a conclusão da Atividade 3.7.

3.8.4 Como:

A disponibilização de um produto de trabalho no repositório de controle de versão gerará uma notificação à equipe de Arquitetura e Qualidade. A partir desta notificação, qualquer dos componentes da equipe poderá proceder a avaliação.

Esta avaliação afere a adequação dos produtos de trabalho, conforme *check-list* previamente definido.

3.8.5 Entradas:

Notificações da disponibilidade de produtos de trabalho no repositório de documentação do projeto.

3.8.6 Saídas:

Nova versão do produto de trabalho (artefato) disponibilizada no repositório de controle de versão, constando resultado da avaliação.

3.8.7 Fluxo:

Se o produto de trabalho (artefato) avaliado estiver em conformidade, o fluxo seguirá para as Atividades 3.10 e 3.11, pois são executadas em paralelo.

Se alguma não conformidade for detectada, o fluxo do processo seguirá para a Atividade 3.9.

3.9 Correção dos Artefatos do 1º Marco

3.9.1 O Que:

Correção de não conformidade(s) apresentada(s) pela equipe de Arquitetura de Qualidade em algum produto de trabalho (artefato).

3.9.2 Quem:

Componente da equipe do projeto responsável pela elaboração do artefato.

3.9.3 Quando:

Após a recepção da informação da existência de não conformidade(s) em algum dos produtos de trabalho (artefatos).

3.9.4 Como:

Editar o artefato e adequá-lo, conforme orientação do analista da qualidade responsável pela avaliação.

3.9.5 Entradas:

Notificação para correção de algum produto de trabalho (artefato).

3.9.6 Saídas:

Nova versão do produto de trabalho (artefato), disponibilizada no repositório de controle de versão da documentação do projeto.

3.9.7 Fluxo:

Retornar ao fluxo da Atividade 3.8, para nova avaliação.

Esta atividade deverá ser repetida enquanto existir alguma não conformidade no artefato.

3.10 Elaboração do *Backlog* da *Sprint*

3.10.1 O Que:

Elaboração do *Backlog* da *Sprint* na ferramenta *Redmine*. Esta elaboração consiste da criação das atividades de análise, desenvolvimento, teste e validação da qualidade.

3.10.2 Quem:

1. Líder da Equipe.
2. Analista de Sistemas.

3.10.3 Quando

Após a *Sprint Planning Meeting*, descrita na Subseção 3.7, deste documento.

3.10.4 Entradas:

Lista de atividades que comporão o *Backlog* da *Sprint*.

3.10.5 Saídas:

As saídas desta atividade são as tarefas identificadas na Tabela 3.7.

Tabela 3.7: Artefatos gerados durante a Atividade 3.10.

Tarefa	Quem?	Prazo
Criação, no *Redmine*[67], das Atividades necessárias para o desenvolvimento da *Sprint*.	Equipe do Projeto	No máximo 4 horas úteis após a receber a lista de estórias.
Planejar a Versão a ser entregue no final da *Sprint* e informar os envolvidos no Projeto.[8]	Equipe do Projeto	Até 1 dia útil após receber as estórias.

3.10.6 Fluxo:

Seguir para a Atividade 3.13.

3.11 Solicitar Definição de Arquitetura

3.11.1 O Que:

Solicitar ao arquiteto alocado ao projeto a definição da Arquitetura.

3.11.2 Quem:

Líder da Equipe.

3.11.3 Quando:

Após a *Sprint Planning Meeting*, descrita na Subseção 3.7, deste documento.

3.11.4 Como:

Através do relato de atividade no *Redmine*.

3.11.5 Entradas:

Documento de Visão.[9]

3.11.6 Saídas:

As saídas desta atividade, são as tarefas descritas na Tabela 3.8.

Tabela 3.8: Tarefas definidas para a Atividade 3.11.

Tarefa	Quem?	Prazo
Atividade relatada no *Redmine*, informando a URL do repositório onde se encontra o documento de visão do projeto.	Líder da Equipe.	Após a entrega do documento pelo Analista.

3.11.7 Fluxo:

Seguir para a Atividade 3.12.

6 Todos os documentos deverão ser armazenados no repositório do projeto.
7 Os documentos das *Sprints* (que contém as Estórias/Requisitos) deverão ser armazenados no repositório de versionamento do projeto separado por *Sprint* (ver exemplo na URL https://github.com/goias/docs).
8 A Versão deverá ser definida pelo Supervisor e a equipe técnica.
9 Ver http://git.intra.goias.gov.br/templates/docs.

3.12 Definição e Documentação da Arquitetura

3.12.1 O Que:

Nesta atividade, o Arquiteto de Software da aplicação deverá iniciar a criação do Documento de Arquitetura e fazer o *commit* no repositório do projeto um documento da *Minimum Viable Architecture* (MVA) e informar o Líder da Equipe.

3.12.2 Quem:

Arquiteto de Software do Projeto.

3.12.3 Quando:

Imediatamente após a Solicitação para a definição da Arquitetura, descrita na Subseção 3.11, deste documento.

3.12.4 Como:

1. Definição da Plataforma de desenvolvimento do Projeto.
2. Elaboração do Documento da Arquitetura do Projeto.

3.12.5 Entradas:

Documento de Visão.

3.12.6 Saídas:

As saídas desta atividade são os artefatos descritos na Tabela 3.9 e as tarefas descritas na Tabela 3.9.

Tabela 3.9: Artefatos gerados durante a Atividade 3.10.

O que?	Quem?	Prazo
Documento MVA	Arquiteto de Software	4 horas úteis após receber a solicitação.
Templates	Arquiteto de Software	4 horas úteis após receber a solicitação.

Tabela 3.10: Tarefas definidas para a Atividade 3.12.

O que?	Quem?	Prazo
Criação do ambiente de desenvolvimento com as ferramentas necessárias.[10]	Arquiteto de Software	1 dia útil.
Realizar reunião para repasse de detalhes do projeto à Infraestrutura.	Arquiteto de Software	3 dias úteis.

3.12.7 Fluxo:

A conclusão desta Atividade é classificada como o segundo Marco do Projeto. Desta forma, o fluxo deve seguir para a Atividade 3.13.

10 O Arquiteto de Software deve providenciar, se necessário, treinamento (*Coding Dojo*) na arquitetura definida.

3.13 Validação dos Artefatos do 2º Marco.

3.13.1 O que:

Validação dos artefatos elaborados durante este fluxo.

3.13.2 Quem:

Analista de Qualidade

3.13.3 Quando:

Após a definição do *Backlog* da *Sprint* e da Definição da Arquitetura.

3.13.4 Como:

Aferir se o(s) artefato(s) estão em conformidade com o PDMS, fazendo o uso do *checklist* de qualidade previamente definido.

3.13.5 Entradas:

Artefatos descritos nas Tabelas 3.7 e 3.9, deste documento.

3.13.6 Saídas:

Nova versão do artefato avaliado, no repositório de controle de versão da documentação do projeto.

3.13.7 Fluxo:

Se os artefatos avaliados não possuírem não conformidades, segue-se para o Fluxo de Desenvolvimento. Se não, segue-se para a Atividade 3.14.

3.14 Correção dos Artefatos do 2º Marco.

3.14.1 O Que:

Correção de não conformidades identificadas nos artefatos do 2º Marco do projeto.

3.14.2 Quem:

Componente da Equipe responsável pela elaboração do artefato.

3.14.3 Quando:

Após a notificação da existência de não conformidade.

3.14.4 Entradas:

Atividade no *Redmine* informando da não conformidade.

3.14.5 Saídas:

A saída identificada para esta atividade encontra-se descrita na Tabela 3.11.

Tabela 3.11: Tarefas definidas para a Atividade 3.14.

Tarefa	Quem?	Prazo
Nova versão do artefato corrigido, disponibilizada no repositório do controle de versão da documentação do projeto.	Analista de Qualidade	1 dia útil.

3.14.6 Fluxo:

Retornar à Atividade 3.13 para nova avaliação.

É importante destacar que o fluxo apresentado nesta Seção possui a iteratividade como característica, de forma que não é necessário terminar todas as atividades e elaborar todos os artefatos antes de se passar para as próximas atividades. Por exemplo, a medida que um artefato é disponibilizado no repositório do controle de versão, o mesmo já será avaliado pela equipe de Arquitetura e Qualidade, independentemente da existência ou não de outros artefatos para a Sprint corrente. E, à medida que a equipe de Arquitetura e Qualidade libera um artefato, o desenvolvimento já pode ser iniciado.

4 Fluxo de Solicitações e Demandas[11] – Manutenção Evolutiva

Um outro Fluxo definido pelo PDMS é o de Solicitações e Demandas para uma Manutenção Evolutiva, que inclui melhorias, consultas e a adição de novas funcionalidades. A representação visual é apresentada no Diagrama 3. O conjunto de atividades e tarefas componentes deste fluxo estão descritas a partir da Subseção 4.1.

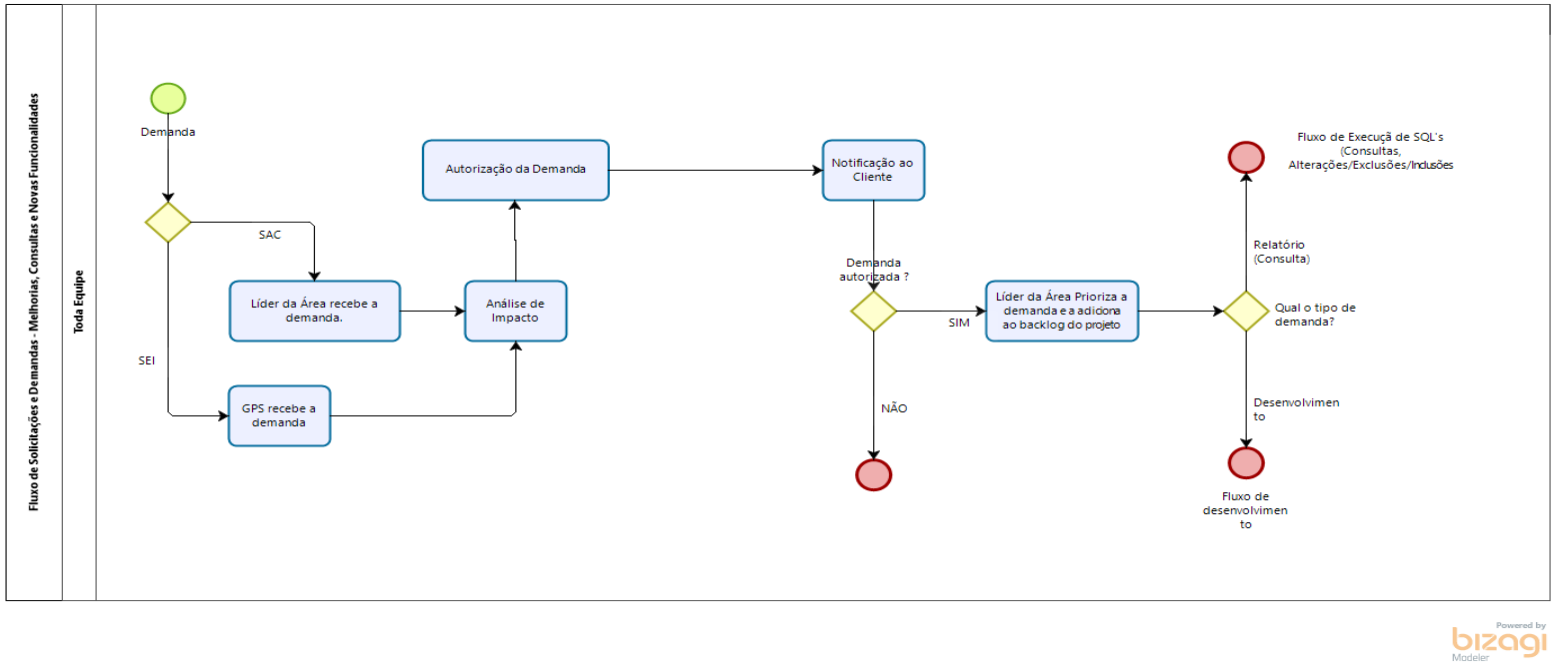

Diagrama 3 - Fluxo de Solicitações e Demandas – Manutenção Evolutiva.

11 Demanda chega via SAC/SEI.

4.1 Líder da Equipe Recebe a Demanda.

4.1.1 O Que:

Demanda apresentada pelo cliente relatando alguma necessidade, que não seja a adição de novas funcionalidades, em relação a algum sistema da GPS.

4.1.2 Quem:

Líder da Equipe.

4.1.3 Quando:

Quando o cliente identificar a necessidade de alguma manutenção e solicitá-la à GPS.

4.1.4 Como:

A demanda é apresenta, através do SAC, que será direcionada para a Fila de Atendimento “GPS – Líderes de Equipe”.

O Líder da Equipe identificará que se trata de uma demanda relativa a algum sistema sob sua coordenação. A partir de então iniciará o processo de atendimento da mesma.

4.1.5 *Entradas*:

Solicitação via SAC.

4.1.6 Saídas:

Esta atividade possui apenas uma tarefa identificada como saída, conforme descrita na Tabela 4.1.

Tabela 4.1: Tarefas definidas para a Atividade 4.1.

O que?	Quem?	Prazo
Demanda identificada e catalogada.	Líder da Equipe	Prazo definido internamente.

4.1.7 Fluxo:

Seguir para a Atividade 4.3.

4.2 Gerente de Projetos e Sistemas Recebe a Demanda

4.2.1 O Que:

Demanda apresentada pelo cliente relatando alguma necessidade da adição de novas funcionalidades, em algum sistema da GPS.

4.2.2 Quem:

O Gerente de Projetos e Sistemas recebe a demanda via SEI.

4.2.3 Quando:

Quando o cliente identificar a necessidade e solicitá-la à GPS.

4.2.4 Como:

A demanda é apresenta, através do SEI, que será direcionada para o Gerente de Projetos e Sistemas.

4.2.5 Entradas:

Solicitação apresentada através do SEI, junto com os documentos necessários: Ofício, memorando, etc.

4.2.6 Saídas:

A saída para esta atividade é a mesma descrita na Subseção 4.1.6.

4.2.7 Fluxo:

Seguir para a Atividade 4.3.

4.3 Análise de Impacto

4.3.1 O Que:

Análise efetuada para determinar:

1. Escopo da demanda.
2. Complexidade da demanda.
3. Impacto no projeto.
4. Disponibilidade de recursos para o atendimento.
5. Tempo de entrega.

4.3.2 Quem:

Líder da Equipe e equipe responsável pelo projeto.

4.3.3 Quando:

Após a recepção da demanda.

4.3.4 Como:

Reunião da equipe para a analisar a demanda, conforme descrito na Subseção 4.3.1.

4.3.5 Entradas:

Demanda apresentada pelo cliente.

4.3.6 Saídas:

O artefato produzido como saída desta atividade está descrito na Tabela 4.2.

Tabela 4.2: Tarefas definidas para a Atividade 4.3.

Tarefa	Quem?	Prazo
Relatório de Análise de Impacto	Líder da Equipe	Prazo definido internamente.

4.3.7 Fluxo:

Seguir para a Atividade 4.4.

4.4 Autorização da Demanda

4.4.1 O Que:

Autorização para atendimento da demanda.

4.4.2 Quem:

1. Gerente de Projetos e Sistemas.
2. Líder da Equipe.

4.4.3 Quando:

Após a conclusão da análise de impacto, descrita na Atividade 4.3.

4.4.4 Como:

Reunião entre o Líder da Equipe e o Gerente de Projetos e Sistemas para a autorização ou não da demanda.

4.4.5 *Entradas*:

Relatório de Análise de Impacto.

4.4.6 Saídas:

Decisão tomada em relação à demanda, que pode ser “Autorizada” ou “Reprovada”.

4.4.7 Fluxo:

Seguir para a Atividade 4.5.

4.5 Notificar o Cliente

4.5.1 O Que:

Notificação ao cliente sofre o resultado da avaliação da demanda apresentada pelo cliente.

4.5.2 Quem:

Gerente de Projetos e Sistemas e Líder da Equipe.

4.5.3 Quando:

Após a tomada de decisão descrita na Atividade 4.4.

4.5.4 Como:

1. O Gerente de Projetos e Sistemas responde ao processo no SEI, informando da autorização

ou não da demanda.

2. O Líder da Equipe responde ao SAC do cliente informando da autorização ou não da demanda.

4.5.5 Entradas:

Saídas da Atividade 4.4, descritas na Subseção 4.4.6.

4.5.6 Saídas:

- Se a demanda foi autorizada, seguir para a Atividade 4.6.
- Se não foi, encerrar o Fluxo.

4.5.7 Fluxo:

4.6 Priorização da Demanda e Adição ao *Backlog* do Projeto.

4.6.1 O Que:

Nesta atividade o Líder da Equipe define a prioridade da demanda e a adiciona ao Backlog do Projeto.

4.6.2 Quem:

Líder da Equipe do Projeto.

4.6.3 Quando:

Após a notificação ao cliente, descrita na Atividade 4.5, deste documento.

4.6.4 Como:

1. Criação de Tarefa(s) no *Redmine*.
2. Planejamento da versão em que o resultado da demanda será entregue ao cliente.
3. Adição da Tarefa ao *Backlog* do Projeto.

4.6.5 *Entradas*:

- Se existir tarefa no *Redmine*, nota com autorização, escopo e prazo (quando for o caso) com o Memorando ou Ofício anexo, se houver.
- Se não existir tarefa no *Redmine*, SAC com nota autorizando bem como escopo, prazo (quando for o caso) e Memorando ou Ofício anexo, se houver.

4.6.6 Saídas:

As saídas desta atividade são as tarefas descritas na Tabela 4.3.

Tabela 4.3: Tarefas definidas para a Atividade 4.6.

Tarefa	Quem?	Prazo

Demanda cadastrada e planejada no *Redmine*.	Líder da Equipe	1 hora após a autorização da demanda.

4.6.7 Fluxo:

Se a Estória/Requisito tratar de uma extração de dados da base de produção (relatórios), seguirá para o [[Fluxo de Extração de Relatórios]]. Senão seguirá para o [[Fluxo de Desenvolvimento]].[12]

12 Ver http://git.intra.goias.gov.br/templates/docs.

5 Fluxo de Solicitações e Demandas – Manutenção Corretiva.

Outro fluxo definido no PDS é o de Solicitações e Demandas para Correções no Sistema, representado visualmente no Diagrama 4. A descrição de cada uma das fases, junto com seus responsáveis e tarefas, é apresentada a partir da Subseção 5.1.

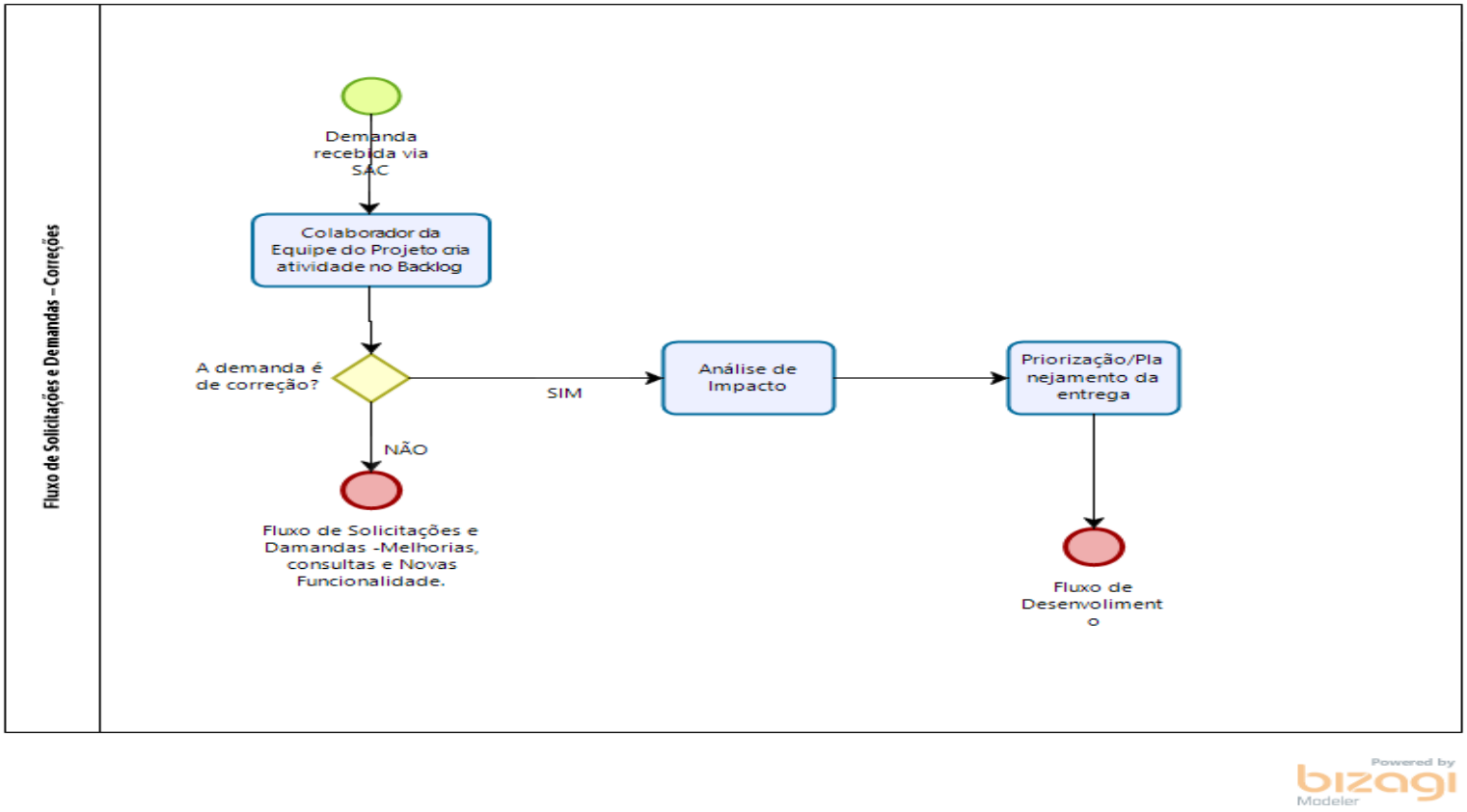

Diagrama 4 - Fluxo de Solicitações e Demandas – Manutenção Corretiva.

5.1 Colaborador da Equipe cria atividade(s) no *Backlog*

5.1.1 O Que:

Formalização da recepção da demanda apresentada pelo cliente através do SAC.

5.1.2 Quem:

Qualquer dos componentes da equipe do projeto. Se nenhum dos colaboradores se prontificar a atender a demanda, o Líder da Equipe deverá atribuir a algum, considerando a urgência da mesma.

5.1.3 Quando:

À medida que existirem demandas no *Backlog* do projeto.

5.1.4 Como:

Acessar o *Backlog* do projeto na ferramenta SAC, atribuir a solicitação e iniciar o trabalho de atendimento.

5.1.5 *Entradas*:

Solicitação via SAC com o Memorando ou Ofício anexados, se houver.

5.1.6 Saídas:

As saídas desta atividade são as Tarefas identificadas na Tabela 13.

Tabela 13: Tarefas definidas para a Atividade 5.1.

Tarefa	Quem?	Prazo
Criar atividade, no *Redmine*, a partir do SAC.	Colaborador/Líder da Equipe.	1 hora após receber a demanda.
Criar as demais atividades (análise, documentação, implementação, teste, etc.), no *Redmine*. Estas atividades serão filhas da atividade criada através do SAC.[13]	Colaborador/Líder da Equipe.	1 hora após receber a demanda.

5.1.7 Fluxo:

- Se a demanda realmente for de correção, ir para a Atividade 5.3.
- Se a demanda não for de correção, ir para o Fluxo de Manutenção Evolutiva.

5.2 Análise de Impacto

A análise de impacto para uma manutenção corretiva é igual àquela definida para uma manutenção evolutiva, conforme descrito na Subseção 4.3, deste documento.

13 Todas as atividades deverão **obrigatoriamente** ter como observadores o Arquiteto de Software, o Administrador de Dados e o Analista de Qualidade pelo projeto seguindo as regras do http://gps.intra.goias.gov.br/index.php/artigos/15-documentacao-Redmine.

5.3 Priorização/Planejamento da Entrega

5.3.1 O Que:

Planejamento de quando a demanda será entregue ao cliente.

5.3.2 Quem:

1. Líder da Equipe.
2. Colaborador alocado para o atendimento.

5.3.3 Quando:

Após a criação da atividade no *Backlog* do projeto, conforme descrito na Subseção 5.1.

5.3.4 *Entradas*:

Atividade (s) desmembrada (s) no *Redmine*.

5.3.5 *Saídas:*

As tarefas identificadas na Tabela 15 são as saídas desta atividade.

Tabela 15: Tarefas definidas para a Atividade 5.3.

Tarefa	Quem?	Prazo
Nova versão do projeto, planejada no *Redmine*, para a entrega da demanda.	Líder da Equipe, colaborador.	Após a elaboração da análise de impacto.

5.3.6 Fluxo:

Seguir para o [[Fluxo de Desenvolvimento]].

6 Fluxo de Desenvolvimento

A representação visual do Fluxo de Desenvolvimento encontra-se no Diagrama 5 e a descrição das atividades associadas são descritas a partir da Subseção 6.1. Vale ressaltar que se a demanda for correção impeditiva/urgente, entra como correção emergencial (*bug-fix*) e segue para *deploy* assim que possível. Caso contrário, entra no *deploy* no final da *Sprint* corrente (se houver).

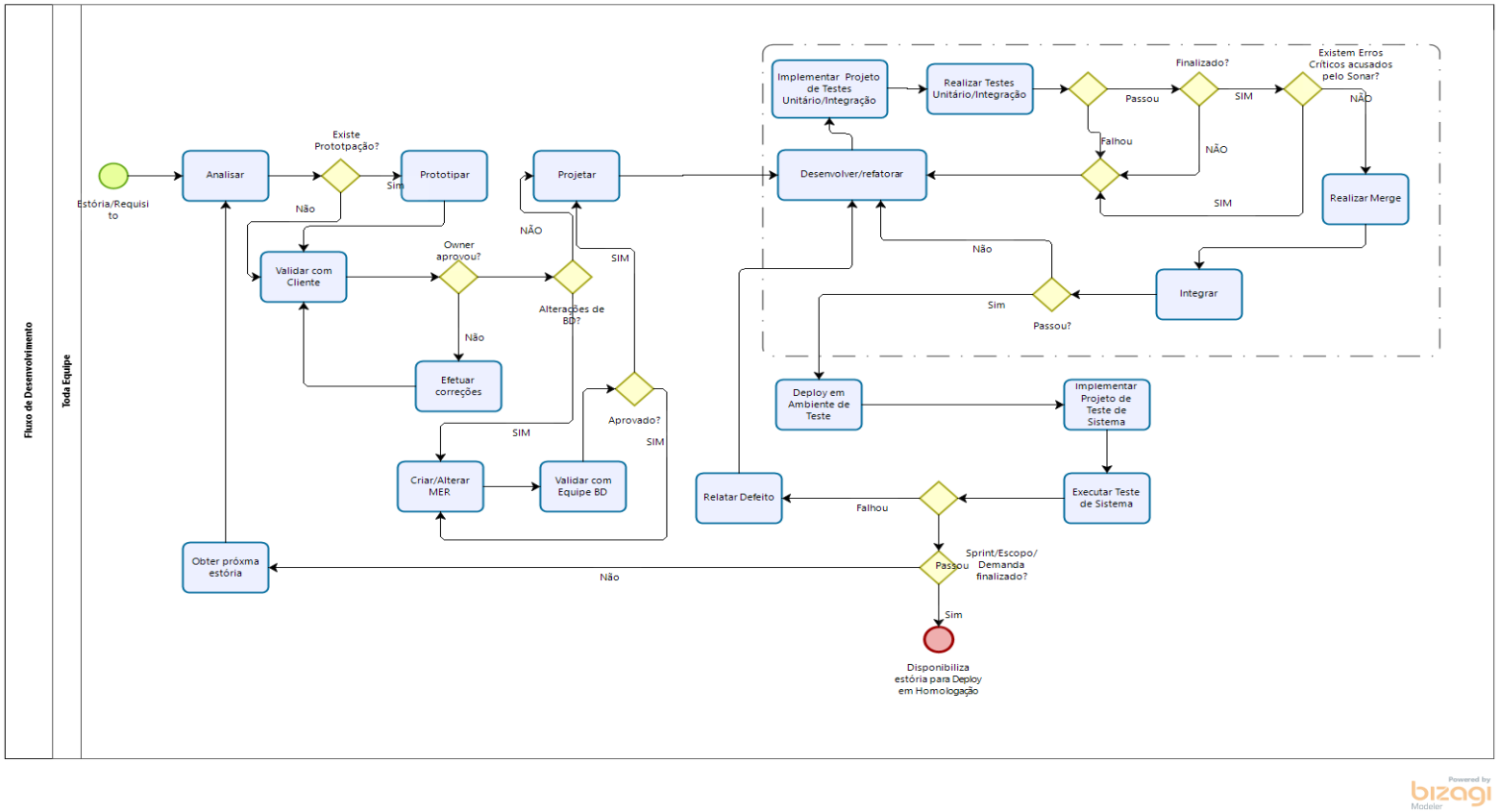

Diagrama 5 – Fluxo de Desenvolvimento

6.1 Analisar

6.1.1 O Que:

Na Atividade de Análise será feito um estudo sobre a Estória/Requisito para subsidiar a sua documentação. Também serão levantados/documentados os Casos de Teste necessários para se validar a Estória/Requisito.

6.1.2 Quem:

1. Analista de Sistemas.
2. Analista de Testes.
3. Administrador de Dados.[14]
4. Arquiteto de Software.[15]

6.1.3 Quando:

Esta atividade é executada sempre que uma demanda, independentemente do seu tipo, esteja liberada para o desenvolvimento.

6.1.4 *Como:*

Esta atividade é elaborada sempre em conjunto com o cliente/usuário. Esta interação com o cliente/usuário pode se dar tanto através de reuniões quando de outras ferramentas de comunicação como e-mail, chat corporativo (*Skype*), chat pessoal (*telegram*, *whatsApp*), etc.

6.1.5 *Entradas:*

Estória/Requisito relatada no *Redmine*.

6.1.6 *Saídas:*

As saídas destas atividades são os artefatos descritos na Tabela 6.1 .

Tabela 6.1: Artefatos definidos para a Atividade 6.1.

Artefato	Quem?	Prazo
Documento de Requisito.[16]	Analista	4 horas
Documento de Visão, caso não exista.[17]	Analista	1 dia útil

6.1.7 Fluxo:

Se existir a necessidade de prototipação, seguir para a Atividade 6.2. Se não existir esta

14 A participação do Administrador de Dados só se fará obrigatória quando houver mudança no modelo de dados. E para isso, o Analista de Sistemas deverá verificar antecipadamente.

15 A participação do Arquiteto de Software só se fará obrigatória quando o desenvolvimento da demanda impactar nos requisitos não funcionais e/ou na arquitetura do projeto.

16 Para uma nova funcionalidade, inicialmente o Analista deverá entregar um documento de requisitos prévio, denominado **Minimum Viable Requirement – MVR (ver http://git.intra.goias.gov.br/templates/docs)** - que conterá a **descrição do requisito e o protótipo** para que o desenvolvedor e o arquiteto possam adiantar o projeto e o desenvolvimento da funcionalidade. Caso seja alteração de uma funcionalidade existente e o documento não exista, o mesmo deve ser criado uma tarefa de débito técnico deve ser criada no backlog para posterior documentação. Se o documento existir, o mesmo deve ser alterado, se for necessário.

17 Ver http://git.intra.goias.gov.br/templates/docs.

necessidade, seguir para a Atividade 6.3.

6.2 Prototipação

6.2.1 O Que:

Elaborar Protótipo de Tela(s) necessárias à implementação da Estória/Requisito em análise.

6.2.2 Quem:

Analista de Sistemas.

6.2.3 Quando:

Durante a atividade de análise de uma Estória/Requisito.

6.2.4 Como:

Utilizando ferramenta de modelagem gráfica e interagindo com o cliente.

6.2.5 Entradas:

Estória/Requisito especificado.

6.2.6 Saídas:

As saídas desta atividade estão descritas nas Tabelas 6.2 e 6.3.

Tabela 6.2: Tarefas definidas para a Atividade 6.2.

Tarefa	Quem?	Prazo
Criar/alterar protótipos/tela(s), o desenho do protótipo da tela.[18]	Analista/ Desenvolvedor	1 dia útil

Tabela 6.3: Artefatos definidos para a Atividade 6.2.

Artefato	Quem?	Prazo
Protótipos das Telas da Estória/Requisito.	Analista	1 hora

6.2.7 Fluxo:

Seguir para a Atividade 6.3.

6.3 Validação com Cliente.

6.3.1 O Que:

Validação da especificação da Estória/Requisito com o cliente.

6.3.2 Quem:

Analista de Sistemas.

18 Caso a Estória/Requisito seja de criação/alteração em telas, um **protótipo deverá ser criado/alterado e aprovado junto ao usuário**. O protótipo da tela deverá ser **preferencialmente** feito junto ao desenvolvedor a fim de usar a tecnologia que será usada para desenvolver a tela. Se não for possível, usar a ferramenta *Pencil*.

6.3.3 Quando:

Sempre que concluir a especificação de uma Estória/Requisito.

6.3.4 Como:

Reunião com cliente para validar a documentação da Estória/Requisito, junto com o respectivo protótipo, quando existir.

6.3.5 Entradas:

1. Documento de especificação da Estória/Requisito.
2. Protótipo.

6.3.6 Saídas:

As saídas desta atividade estão descritas na Tabela 6.4.

Tabela 6.4: Tarefas definidas para a Atividade 6.3.

Tarefa	Quem?	Prazo
Email do cliente informando a aprovação dos artefatos avaliados.[19]	Cliente	No encerramento da seção de validação.
SAC solicitando a correção das não conformidades identificadas.	Cliente	No encerramento da seção de validação.

6.3.7 Fluxo:

Caso alguma não conformidade for detectada, o fluxo segue para a Atividade 6.4.

Se estiver tudo OK, avaliar se há a necessidade de se criar/alterar o Modelo de Entidade e Relacionamento – MER. Se sim, o fluxo segue para a Atividade 6.5. SE não, o fluxo segue para a Atividade 6.8.

6.4 Efetuar Correções nos Documentos de Requisitos e Protótipos

6.4.1 O Que:

Eliminar as não conformidades identificadas pelo cliente.

6.4.2 Quem:

Analista de Sistemas.

6.4.3 Quando:

Após a Validação com o cliente, descrita na Subseção 6.3, deste documento.

6.4.4 Como:

Editando e alterando os artefatos em que foram identificadas as não conformidades.

19 Uma cópia deste e-mail deverá ser guardada no repositório de controle de versão da documentação do projeto.

6.4.5 Entradas:

SAC criado pelo cliente solicitando a correção das não conformidades identificadas.

6.4.6 Saídas:

A saída desta atividade está descrita na Tabela 6.5.

Tabela 6.5: Tarefas definidas para a Atividade 6.4.

Tarefa	Quem?	Prazo
Nova versão do(s) artefato(s) em que a(s) não conformidade(s) foi(ram) identificada(s), disponibilizada no repositório de controle de versão da documentação do projeto.	Analista de Sistemas	1 dia útil.

6.4.7 Fluxo:

Retornar à Atividade 6.3 para nova validação com o cliente.

6.5 Criar/Alterar MER

6.5.1 O Que:

Elaborar o Modelo de Dados e Relacionamento – MER.

6.5.2 Quem:

Analista de Sistemas.

6.5.3 Quando:

Após a validação com o cliente dos artefatos de especificação da Estória/Requisito, conforme descrito na Atividade 6.3, deste documento.

6.5.4 Como:

1. Se for um projeto novo:

   1.1. Solicitar, através do SAC, a criação do contexto para o projeto no Banco de Dados.

   1.2. Elaborar o MER utilizando a ferramenta *Oracle Designer*.

2. Se for um projeto existente, abrir e editar o MER existente, utilizando o *Oracle Designer*.

6.5.5 Entradas:

Artefatos elaborados durante a especificação da Estória/Requisito.

6.5.6 Saídas:

A saída desta atividade está identificada na Tabela 6.6.

Tabela 6.6: Tarefas definidas para a Atividade 6.5.

Tarefa	Quem?	Prazo
Modelo de Entidade e Relacionamento	Analista de Sistemas	1 dia útil.

6.5.7 Fluxo:

Seguir para a Atividade 6.6.

6.6 Validar MER com BD

6.6.1 O Que:

Validação do Modelo de Entidades e Relacionamento com a Equipe de Banco de Dados.

6.6.2 Quem:

1. Analista de Sistemas.
2. Analista de Dados (AD).

6.6.3 Quando:

Imediatamente após o encerramento da especificação da Estória/Requisito.

6.6.4 Como:

Reunião com o Analista de Dados, membro da equipe de Banco de Dados. Nesta reunião serão discutidas, questões relativas à:

1. Exatidão e completude do modelo, à luz dos padrões definidos pela Equipe de Banco de Dados.
2. Relacionamentos com objetos existentes em outros sistemas.

6.6.5 Entradas:

MER.

6.6.6 Saídas:

As saídas definidas para esta atividade estão descritas na Tabela 6.7.

Tabela 6.7: Tarefas definidas para a Atividade 6.6.

Tarefa	Quem?	Prazo
Relatório de Correções necessárias sobre o modelo. No cenário em que haverá necessidade de correções.	Analista de Dados	1 dia útil.
E-mail atestando a aprovação do Modelo. No cenário em que não haverá a necessidade de correções.	Analista de Dados	1 dia útil.

6.6.7 Fluxo:

1. Havendo a necessidade de correção, o fluxo volta para a Atividade 6.5.
2. Não havendo tal necessidade, o fluxo segue para a Atividade 6.7.

6.7 Projetar

6.7.1 O Que:

Nesta atividade será discutido junto ao Arquiteto de Software do Projeto a melhor (ou a mais viável) solução técnica para atender à Estória/Requisito.

6.7.2 Quem:

1. Analista de Sistemas.
2. Desenvolvedor/Implementador.
3. Arquiteto de Software.
4. Administrador de Dados (Opcional).

6.7.3 Quando:

Após a aprovação do MER, pela equipe de Banco de Dados

6.7.4 Como:

Reunião entre os envolvidos na atividade para definir a melhor estratégia de implementação da solução proposta pelo Analista de Sistemas.

6.7.5 *Entradas:*

- Documento de Especificação da Estória/Requisito.[20]
- Se envolver criação/alteração em telas, o protótipo da tela (ver na Atividade 6.2) aprovado pelo usuário.
- Documento de Arquitetura.[21]

6.7.6 *Saídas:*

Os artefatos descritos na Tabela 6.8 são as saídas desta atividade.

Tabela 6.8: Artefatos definidos para a Atividade 6.7.

Artefato	Quem?	Prazo
Diagrama de sequência (Opcional)	Analista	4 horas
Diagrama de classes (Opcional)	Analista	4 horas

20 Ver https://github.com/goias/docs. Inicialmente o Analista de Sistemas só entregará o MVR (ver Atividade 6.1) a fim de agilizar o desenvolvimento.

21 Ver http://git.intra.goias.gov.br/templates/docs. Se for a primeira Sprint de um projeto novo o documento deve ser o Minimum Viable Architecture – MVA.

Solução técnica em formato descritivo em atividade no *Redmine*.[22] [23]	Analista/ Desenvolvedor	4 horas
Atualização do Documento de Arquitetura (caso necessário)	Arquiteto de Software.	1 hora.

6.7.7 Fluxo:

Após projeto criado e aprovado pelo Arquiteto de Software, seguir para a Atividade 6.3.

6.8 Ciclo de desenvolvimento

6.8.1 Desenvolver/Refatorar

6.8.1.1 O que:

Atividade em que a criação/alteração de código-fonte deverá ser efetivada.

6.8.1.2 Quem:

1. Desenvolvedor.

6.8.1.3 Quando:

Após a elaboração do projeto para o atendimento à demanda, descrito na Subseção 6.2.

6.8.1.4 Como:

Traduzindo para a linguagem de programação do projeto, a especificação da Estória/Requisito. Observando a solução técnica proposta pelo Arquiteto de Software.

6.8.1.5 Entradas:

- Documento de Especificação da Estória/Requisito.
- Modelo de Entidades e Relacionamentos – MER.
- Diagrama de sequência (Opcional).
- Diagrama de classes (Opcional).
- Solução técnica em formato descritivo em atividade no *Redmine*.

6.8.1.6 Saídas:

Os artefatos identificados na Tabela 6.9 são as saídas desta atividade.

Tabela 6.9: Artefatos definidos para a Atividade 6.8.1.

Artefato	Quem?	Prazo
Código fonte “executável” (i.e. sem erros de compilação) para a Estória/Requisito, disponibilizado no repositório de controle de versão do projeto.	Desenvolvedor	N/A

22 Quando a demanda se tratar de nova funcionalidade, o arquiteto deverá aprovar o projeto através de nota na atividade no Redmine atestando a solução.

23 Reuniões/conversas diárias deverão ser provocadas pelo Arquiteto de Software do projeto a fim de alinhar problemas e soluções técnicas a serem aplicadas ao mesmo. Revisões de código também deverão ser feitas a fim de mitigar problemas (qualidade de código, não atendimento de requisitos não funcionais etc.) futuros.

6.8.1.7 Fluxo:

Seguir para a Atividade 6.8.2.

6.8.2 Implementar Projeto de Teste Unitário/Integração.

6.8.2.1 O Que:

Atividade em que os casos de testes unitários e de integração serão efetivamente implementados, na mesma linguagem de programação utilizada no desenvolvimento do projeto.

Vale ressaltar que o teste unitário tem como escopo exclusivamente os métodos de uma classe específica e o teste de integração tem seu foco voltado para a interação entre os elementos componentes do sistema, tais como: associação entre classes, comunicação com APIs, interação com o Banco de Dados, etc. Este dois níveis de testes são implementados em um único projeto de teste, junto com o projeto de desenvolvimento da aplicação.

6.8.2.2 Quem:

Desenvolvedor.

6.8.2.3 Quando:

Após a conclusão da implementação da Estória/Requisito, descrita na Subseção 6.8.1, deste documento.

6.8.2.4 Como:

Traduzindo para a linguagem de programação do projeto, a especificação dos casos de testes elaborados pelo Analista de Teste. Utilizando um *framework* próprio para esta modalidade de teste. Alguns exemplos destes *frameworks* são: JUnit, NUnit e PHPUnit.

6.8.2.5 Entradas:

Especificação dos casos de testes (unitários/integração). Esta especificação dos casos de teste deverá ser elaborada utilizando a Ferramenta *Testlink*.

6.8.2.6 Saídas:

Tabela 6.10: Artefatos definidos para a Atividade 6.8.2.

Artefato	Quem?	Prazo
Código fonte “executável” (i.e. sem erros de compilação) para os testes unitários/integração da Estória/Requisito, disponibilizado no repositório de controle de versão do projeto.	Desenvolvedor	N/A

6.8.2.7 Fluxo:

6.8.3 Executar Teste Unitário/Integração

6.8.3.1 O Que:

Atividade em que os testes unitários/integração deverão ser executados a fim de garantir o correto funcionamento de cada um dos objetos implementados, conforme descrito na Subseção 6.8.1.

6.8.3.2 Quem:

Este teste será executado automaticamente pela ferramenta de gerenciamento do projeto. Esta execução será disparada pontualmente pelo desenvolvedor ou totalmente automatizada no momento do *build* de alguma versão do projeto.

6.8.3.3 Quando:

Após a atividade de implementação do projeto de testes unitários/integração, descrita na Subseção 6.8.2.

6.8.3.4 Entradas:

1. Código-fonte “executável” do Projeto de Testes Unitários/Integração.

6.8.3.5 Saídas:

O artefato listado na Tabela 6.11 é a saída desta atividade.

Tabela 6.11: Artefatos definidos para a Atividade 6.8.3.

Artefato	Quem?	Prazo
Relatório de Resultado dos testes, disponibilizado no repositório de controle de versão do projeto.	Desenvolvedor	N/A

6.8.3.6 Fluxo:

1. Se os testes não passarem, o ciclo deve voltar para a Atividade 6.8.1.
2. Se os testes passarem e a implementação da Estória/Requisito não estiver finalizada, o ciclo deve voltar para a Atividade 6.8.1.
3. Se os testes passarem, a implementação da Estória/Requisito estiver finalizada e o Sonar detectar a presença de problemas críticos no código fonte, o ciclo deve voltar para a Atividade 6.8.1.
4. Se os testes passarem, a implementação da Estória/Requisito estiver finalizada e o Sonar não detectar a presença de problemas críticos no código fonte, seguir para a Atividade 6.8.4.

6.8.4 Realizar Merge

6.8.4.1 O Que:

Atividade em que os arquivos de código-fonte criados/alterados, em *branch* próprio, deverão ser unidos ao *branch* de desenvolvimento principal (*master/trunk*).

6.8.4.2 Quem:

1. Líder da Equipe, com suporte do Arquiteto de Software.

6.8.4.3 Quando:

Após a atividade de execução dos testes unitários/integração, descrita na Subseção 6.8.3.

6.8.4.4 Como:

Esta tarefa é executada através da ferramenta de controle de versão GitLab.

6.8.4.5 Entradas:

1. Código fonte “executável” (i.e. sem erros de compilação) para a estória/requisito.
2. Relatório de execução dos Testes Unitários/Integração.

6.8.4.6 Saídas:

Os Artefatos identificados na Tabela 6.12 são as saídas desta atividade.

Tabela 6.12: Artefatos definidos para a Atividade 6.8.4.

Artefato	Quem?	Prazo
Arquivos de código-fonte unificados à linha de desenvolvimento principal (*master/trunk*).	Desenvolvedor	N/A

6.8.4.7 Fluxo:

Após realizado o *Merge* e resolvido os possíveis conflitos, seguir para a Atividade 6.8.5.

6.8.5 Integrar

6.8.5.1 O Que:

Nesta atividade o Desenvolvedor deverá submeter os arquivos de código-fonte desenvolvidos à ferramenta de Integração Contínua (Jenkins) para execução automatizada dos Testes de Regressão (Unitário e Integração) de toda a aplicação. Esta atividade é essencial e obrigatório para garantia de qualidade da aplicação.[24]

6.8.5.2 Quem:

Desenvolvedor, com suporte do Arquiteto de Software.

6.8.5.3 Quando:

Após a atividade de *Merge*, descrita na Subseção 6.3.3.

6.8.5.4 Entradas:

1. Código fonte “executável” (i.e. sem erros de compilação) para o requisito.
2. Testes Unitários.
3. Testes de integração.

6.8.5.5 Saídas:

Os artefatos identificados na Tabela 6.13 são as saídas desta atividade.

Tabela 6.13: Artefatos definidos para a Atividade 6.8.5.

Tarefa	Quem?	Prazo
Resultado dos testes	Desenvolvedor	N/A

24 Todos os dias uma *build* será executado às 12:00 e outro às 00:00. Caso necessário, o próprio desenvolvedor executará a tarefa de integração em projetos que precisem de menor tempo de resposta (ie. fora da janela de execução de tarefas). Outros projetos também poderão executar os testes no ambiente de Integração Contínua ao efetuar o *commit* do código-fonte.

6.8.5.6 Fluxo:

Caso os testes no ambiente de Integração Contínua não passem, a equipe é avisada (por *email*) e o ciclo volta para a Atividade 6.8.1, para as devidas correções.

6.9 Realizar *Deploy* no Ambiente de Teste

6.9.1 O Que:

Disponibilização da nova versão do projeto no ambiente de teste.

6.9.2 Quem:

Desenvolvedor utilizando o *Jenkins*.

6.9.3 Quando:

Após a atividade de integração, conforme descrito na Seção 6.8.5, deste documento.

6.9.4 Como:

Esta atividade é um dos passos constantes do pipeline do processo de *deploy*, executado automaticamente pelo *Jenkins*.

6.9.5 Entradas:

Nova versão do projeto preparada para o processo de *deploy*.

6.9.6 Saídas:

A saída desta atividade está descrita na Tabela 6.14.

Tabela 6.13: Artefatos definidos para a Atividade 6.9.1.

Tarefa	Quem?	Prazo
Nova versão do projeto disponível no ambiente de teste.	Desenvolvedor	N/A

6.9.7 Fluxo:

Seguir para a Atividade 6.10.

6.10 Implementar/Atualizar Projeto de Teste de Sistema.

6.10.1 O Que:

Teste de Sistema é um nível de teste em que todo o sistema é testado, preferencialmente de forma automatizada. Desta forma, nesta atividade, os casos de testes especificados para este nível de teste, que poderão/deverão ser executados de forma automatizada, serão implementados na linguagem de programação utilizada na implementação do projeto.

O Teste de Sistema utiliza a técnica de teste funcional e seu objetivo é testar a aplicação como um todo para validar:

1. Se os resultados obtidos a partir da execução de cada uma das funcionalidades disponibilizadas pelo sistema, estão de acordo com os esperados, conforme consta da especificação de requisitos.
2. Se os requisitos e padrões de qualidade estão de acordo com o especificado.

6.10.2 Quem:

Analista de teste alocado ao projeto.

6.10.3 Quando:

Sempre que uma nova versão do projeto esteja disponível no ambiente de teste.

6.10.4 Como:

Se a aplicação ainda não possui um projeto de teste de sistema, este deverá ser criado. Se já possui, deverá ser atualizado de acordo com as necessidades da nova versão disponível para teste. Tanto a criação quanto a atualização é feita através da implementação dos casos de testes na linguagem de programação em que foi implementada a aplicação.

Este projeto utilizará ferramentas próprias para a implementação de testes funcionais automatizados. No momento a principal ferramenta utilizada é o *Selenium Webdriver*.

6.10.5 Entradas:

1. Nova versão da aplicação disponível no ambiente de teste.
2. Especificação de casos de teste funcional, preferencialmente disponível no *Testlink*.

6.10.6 Saídas:

A saída desta atividade está identificada na Tabela 6.14.

Tabela 6.14: Artefatos definidos para a Atividade 6.10.

Tarefa	Quem?	Prazo
Nova Versão do Projeto de Teste de Sistema disponível no repositório do controle de versão.	Analista de Teste	15 dias.

6.10.7 Fluxo:

Seguir para a Atividade 6.11.

6.11 Executar o Teste de Sistema

6.11.1 O Que:

Atividade em que o Teste de Sistema, descrito na Atividade 6.10, é executado.

6.11.2 Quem:

1. O Analista de Testes executará aqueles casos de teste cuja execução foi definida como “Manual”.

2. O Analista de Teste será responsável pela análise do resultado do teste automatizado executado pelo *Jenkins*.

6.11.3 Quando:

Após a conclusão da implementação, descrito na Subseção 6.10.

6.11.4 *Como:*

1. Execução manual – executar a aplicação no ambiente de teste, seguindo o roteiro especificado nos casos de teste documentados no Testlink.
2. Execução automatizada disparado pelo *Jenkins*.

6.11.5 *Entradas:*

1. Nova versão do projeto a ser testada.
2. Lista de Casos de Testes documentados no *Testlink*.

6.11.6 *Saídas:*

O artefato identificado na Tabela 6.15, é a saída desta atividade.

Tabela 6.15: Artefatos definidos para a Atividade 6.11.

Artefato	Quem?	Prazo
Novo relatório de testes disponibilizado no repositório de controle de versão da documentação do projeto.	Analista de Sistemas/Testes	N/A

6.11.7 Fluxo:

1. Caso os testes não passem, ir para a Atividade 6.12.
2. Caso os testes passem e a *Sprint* estiver finalizado, ir para o [Fluxo de *Deploy*].
3. Caso os testes passem e a *Sprint* não estiver finalizado, ir para a Atividade 6.13.

6.12 Relatar Não Conformidade.

6.12.1 O Que:

Relato de não conformidade detectada duranta a execução do teste de sistema.

6.12.2 Quem:

1. Analista de Sistemas relata não conformidade detectada durante a execução manual.
2. Analista de Testes relata não conformidade detectada durante a execução automatizada.

6.12.3 Quando:

Após um ciclo de execução do teste de sistema em que se detecta alguma não conformidade.

6.12.4 Como:

Através de atividade relatada no *Redmine*

6.12.5 Entradas:

Problema encontrado ao testar aplicação.

6.12.6 Saídas:

As saídas desta atividade são as tarefas descritas na Tabela 24.

Tabela 24: Tarefas definidas para o Atividade 6.5.

Tarefa	Quem?	Prazo
Atividade relatando não conformidade no Redmine	Analista de Sistemas/Testes	1 hora.

6.12.7 Fluxo:

1. Se a não conformidade for relativa a problemas na implementação da estória/requisito, o fluxo volta para Atividade 6.8.1.
2. Se a não conformidade for relativa a problemas relativos à especificação do Caso de Teste, o Fluxo volta para a Atividade 6.1.
3. Se a não conformidade for relativa a problemas relativos à implementação do Caso de Teste, o Fluxo volta para a Atividade 6.10.

6.13 Obter próxima Estória/Requisito

6.13.1 O Que:

Iniciar o ciclo de desenvolvimento da próxima estória/requisito, constante do *Backlog* do projeto no *Redmine*.

6.13.2 Quem:

1. Analista de Sistemas.

6.13.3 Quando:

Após a execução bem-sucedida da atividade de teste, descrita na Subseção 6.11.

6.13.4 *Como:*

Analista de Sistema atribui a si a atividade pertinente ao início do desenvolvimento da próxima estória/requisito.

6.13.5 *Entradas:*

Não se aplica.

6.13.6 *Saídas:*

Não se aplica.

6.13.7 Fluxo:

Volta para a Atividade 6.1.

7 Fluxo de Execução de SQL's (Consultas, Alterações/Exclusões/Inclusões)

Uma visão geral deste fluxo é apresentada no Diagrama 6. A descrição das atividades associadas encontra-se neste documento, a partir da Subseção 7.1.

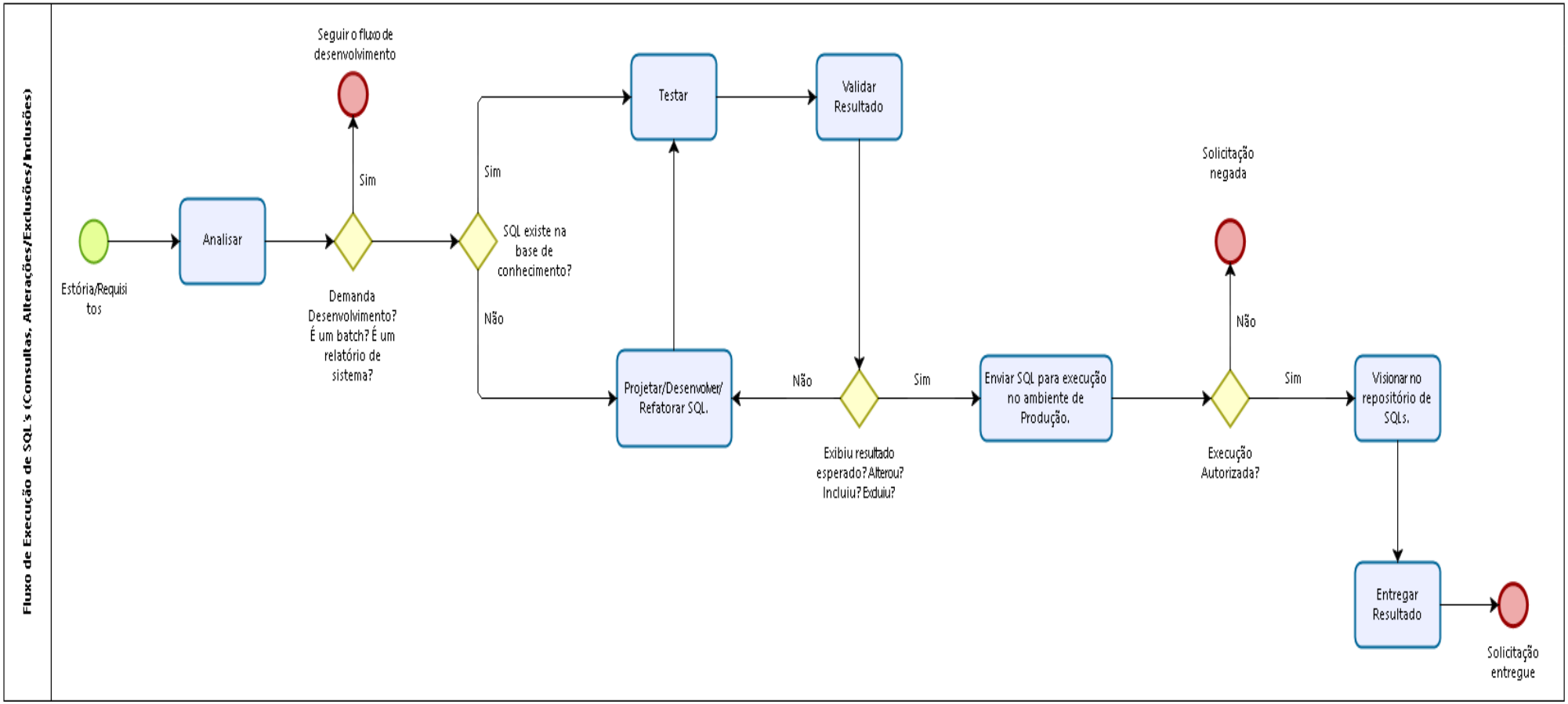

Diagrama 6 - Fluxo de Execução de SQL's (Consultas, Alterações/Exclusões/Inclusões)

7.1 Analisar

7.1.1 O Que:

Nesta atividade será feita uma verificação da viabilidade/necessidade da execução do SQL e tabelas a serem usadas.

7.1.2 Quem:

1. Analista de Sistemas.
2. Administrador de Dados.

7.1.3 Quando:

Sempre que uma demanda desta natureza for autorizada.

7.1.4 *Entradas:*

Solicitação de execução de SQL (Via *Redmine*).[25]

7.1.5 *Saídas:*

As saídas desta atividade são os artefatos descritos na Tabela 7.1.

Tabela 7.1: Artefatos definidos para a Atividade 7.1.

Artefato	Quem?	Prazo
O SQL a ser executado, caso o mesmo exista no repositório	Analista de Sistemas	N/A
Caso o SQL não exista no repositório[26], as tabelas e os critérios de filtros a serem utilizados no SQL	Analista de Sistemas	N/A

7.1.6 *Fluxo:*

1. Se a solicitação for uma demanda de desenvolvimento, ir para o Fluxo de Desenvolvimento.
2. Se o SQL existir no repositório versionamento ir para a Atividade 7.3, senão ir para a 7.2.

7.2 Projetar/Desenvolver/Refatorar SQL

7.2.1 O Que:

No passo do Projeto, o SQL deve ser construído/alterado pensando na melhor solução técnica para atender a solicitação, aplicando-se os filtros necessários.

7.2.2 Quem:

1. Analista Sistemas.
2. Administrador de Dados.

25 Se a solução for para alteração/inclusão/exclusão de dados, o ofício/memorando deve ser obrigatoriamente anexado à solicitação.

26 Ver http://git.intra.goias.gov.br/templates/docs.

3. Administrador de Banco de Dados (Opcional).[27]

7.2.3 Quando:

Após a atividade de análise, descrita na Subseção 7.2.

7.2.4 *Entradas:*

1. Tabelas a serem usadas.
2. Campos a serem retornados (se for o caso).
3. Filtros a serem aplicados.

7.2.5 *Saídas:*

O artefato descrito na Tabela 7.2 é a saída desta atividade.

Tabela 7.2: Artefatos definidos para a Atividade 7.2.

Artefato	Quem?	Prazo
SQL a ser executado no Ambiente de Desenvolvimento.	Analista de Sistemas	N/A

7.2.6 *Fluxo:*

Ir para a Atividade 7.3.

7.3 Testar

7.3.1 O Que:

Analista de Sistemas executa o SQL no Ambiente de Desenvolvimento.

7.3.2 Quem:

Analista de Sistemas.

7.3.3 Quando:

Após a atividade descrita na Subseção 7.2.

7.3.4 *Entradas:*

SQL construído.

7.3.5 *Saídas:*

O artefato descrito na Tabela 7.3 é a saída desta atividade.

Tabela 7.3: Artefato definido para a Atividade 7.3.

Artefato	Quem?	Prazo
Resultado a ser validado	Analista de Sistemas	N/A

27 Opcionalmente um DBA pode ser envolvido para auxiliar em otimização etc.

7.3.6 *Fluxo:*

Ir para a Atividade 7.4.

7.4 Validar Resultado

7.4.1 O Que:

Analista de Sistemas valida junto ao Líder da Equipe o resultado do SQL.[28]

7.4.2 Quem:

1. Analista de Sistemas.
2. Líder da Equipe.

7.4.3 Quando:

Após a Atividade descrita na Subseção 7.3.

7.4.4 Entradas:

Resultado do SQL

7.4.5 *Saídas:*

A tarefa descrita na Tabela 7.4 é a saída desta atividade.

Tabela 7.4: Tarefa definida para a Atividade 7.4.

Artefato	Quem?	Prazo
Resultado da validação (atende ou não atende?)	Analista de Sistemas	N/A

7.4.6 Fluxo:

1. Se o resultado da execução do SQL atender ao esperado ir para a Atividade 7.5.
2. Senão, voltar para a 7.2.

7.5 Enviar SQL para execução em produção

7.5.1 O Que:

Analista de Sistemas envia (via SAC) o SQL para execução no Ambiente de Produção.

7.5.2 Quem:

Analista de Sistemas.

7.5.3 Quando:

Após a execução da atividade descrita na Subseção 7.4.

28 A validação do resultado aqui não se refere aos dados que, obviamente, serão diferentes dos dados de produção mas sim ao filtro, formato, ordem, agrupamento, colunas etc.

7.5.4 Entradas:

SQL finalizado.

7.5.5 Saídas:

A tarefa identificada na Tabela 7.5 é a saída desta atividade.

Tabela 29: Tarefas definidas para a Atividade 7.5.

Artefato	Quem?	Prazo
Solicitação via SAC de execução de SQL no Ambiente de Produção.[29]	Analista de Sistemas	Assim que finalizar o SQL.

7.5.6 Fluxo:

Ir para a Atividade 7.6.

7.6 Versionar no Repositório de SQL's

7.6.1 O Que:

Analista de Sistemas faz *upload* do SQL para o repositório de versionamento seguindo o Padrão de Versionamento de SQL's.[30]

7.6.2 Quem:

Analista de Sistemas.

7.6.3 Quando:

Após a execução da atividade descrita na Subseção 7.6.

7.6.4 Entradas:

SQL finalizado.

7.6.5 *Saídas:*

O artefato descrito na Tabela 7.6 é a saída desta atividade.

Tabela 7.6: Artefatos definidos para a Atividade 7.6.

Artefato	Quem?	Prazo
SQL versionado	Analista de Sistemas	Assim que finalizar o SQL.

7.6.6 Fluxo:

Ir para a Atividade 7.7.

29 O Administrador de Dados do Projeto poderá executar o SQL com suas credenciais.
30 Ver http://git.intra.goias.gov.br/templates/docs.

7.7 Entregar resultado

7.7.1 O Que:

Analista de Sistemas entrega resultado da execução do SQL.

7.7.2 Quem:

Analista Sistemas.

7.7.3 Quando:

Após a execução da atividade descrita na Subseção 7.6.

7.7.4 Entradas:

1. SQL finalizado.
2. Resultado

7.7.5 *Saídas:*

A tarefa descrita na Tabela 7.7 é a saída desta atividade.

Tabela 7.7: Tarefa definida para a Atividade 7.7.

O que?	Quem?	Prazo
Entregar resultado e o SQL usado anexado à atividade solicitante no Redmine	Analista de Sistemas	Assim que o SQL for executado e o resultado apresentado.

7.7.6 Fluxo:

Finaliza a processo.

8 Fluxo de *Deploy*

A representação visual do Fluxo de *Deploy*, pode ser conferida no Diagrama 7. A descrição das atividades encontra-se a partir da Subseção 8.1. Sobre este fluxo é importante ressaltar que todos os passos são de responsabilidade do *Jenkins* e terão a sessão **Tempo de Espera**, que corresponde ao tempo que *Jenkins* aguarda por uma interação do usuário (Analista de Sistemas, Arquiteto de Software, Analista de Qualidade, Líder da Equipe ou Gerente de Projetos e Sistemas) e, caso esse tempo seja alcançado, o processo é automaticamente cancelado.

8.1 Iniciar a *Build*

8.1.1 O Que:

Início da execução das etapas do *pipeline* da tarefa de *deploy* da Versão no Ambiente de Testes.[31]

8.1.2 Quem:

Desenvolvedor/Arquiteto.

8.1.3 Quando:

Sempre que algum ciclo de desenvolvimento/manutenção for concluído e o resultado estiver pronto para ser entregue ao Cliente.

8.1.4 Entradas:

Versão para *deploy* em Testes.

8.1.5 *Saídas:*

Início da execução da atividade pelo *Jenkins*.

8.1.6 Fluxo:

Ir para a Atividade 8.2.

8.2 *Build* e *deploy* em ambiente de testes

8.2.1 O Que:

Jenkins realiza *build* e *deploy* no ambiente de testes.

8.2.2 Quem:

Jenkins.

8.2.3 Quando

Após a execução da Atividade descrita na Subseção 8.1.

8.2.4 Entradas:

Execução da tarefa (ver Atividade 8.1).

31 Antes de executar o *deploy* no ambiente de testes o Analista de Sistemas deve certificar-se que todas as alterações de banco de dados foram replicadas para o banco de dados de testes.

8.2.5 *Saídas:*

Build e *deploy* executado.

8.2.6 Fluxo:

Ir para a Atividade 8.3.

8.3 Validação de *build* e *deploy*

8.3.1 O Que:

Analista verifica se houve algum erro ao realizar a *build* e o *deploy.*

8.3.2 Quem:

Líder da Equipe/Arquiteto de Software.

8.3.3 Quando:

Após a *build* e o *deploy* terem sido feitos no ambiente de testes, conforme descritos na Subseção 8.2**.**

8.3.4 *Entradas:*

Resultado da *build* e do *deploy* no *Jenkins*.

8.3.5 *Saídas:*

Análise realizada

8.3.6 Fluxo:

1. Caso ocorra algum problema na *build ou no deploy*, deve-se voltar [Fluxo de Desenvolvimento] e envolver o arquiteto do sistema na solução do problema.
2. Caso **não** dê problema na *build ou no deploy*, o Analista/Arquiteto deve aprovar o passo no *Jenkins* e o fluxo segue para o Passo 8.4.

8.3.7 Tempo de espera:

Uma hora.

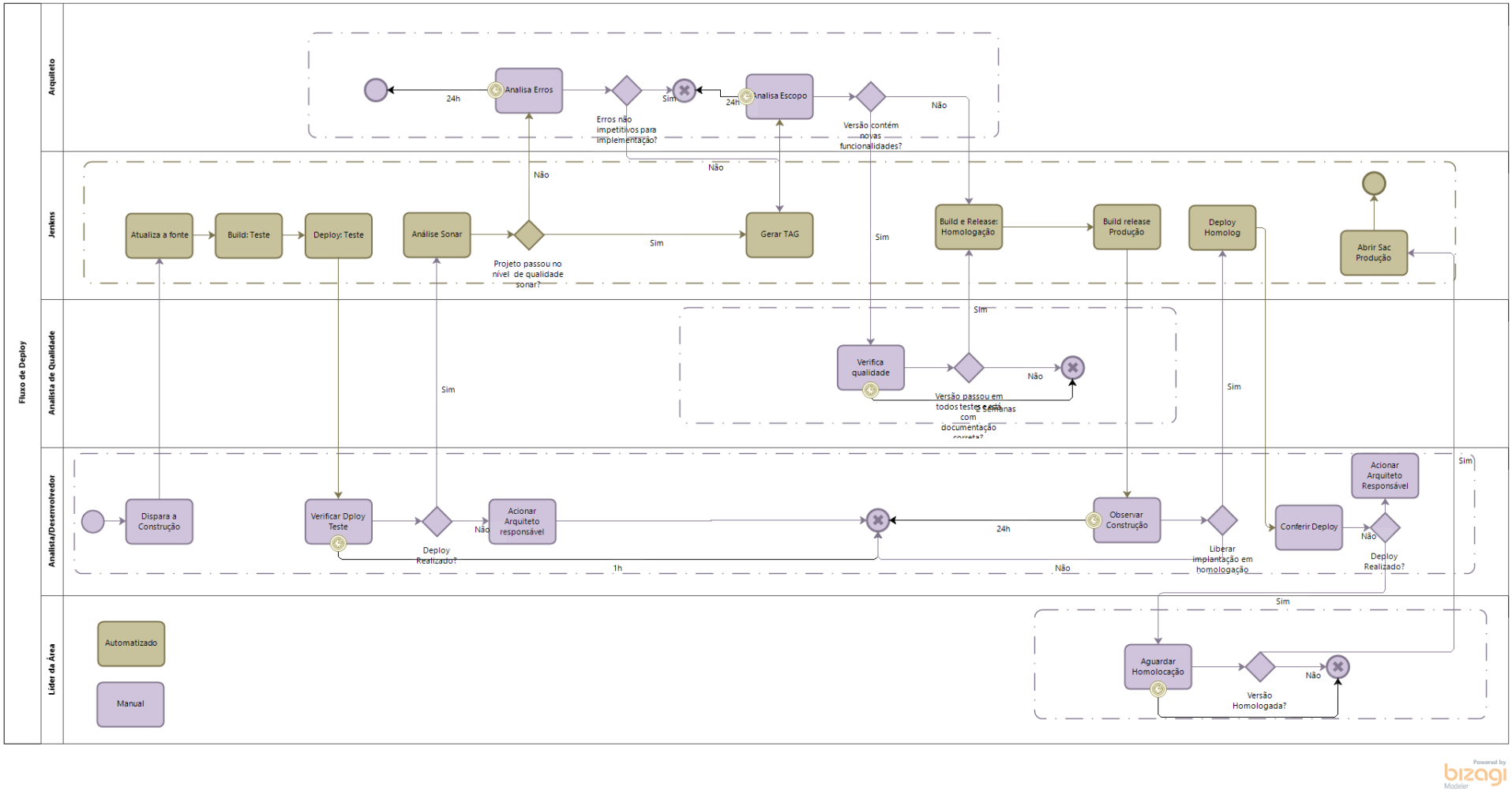

Diagrama 7 – Fluxo de *Deploy*

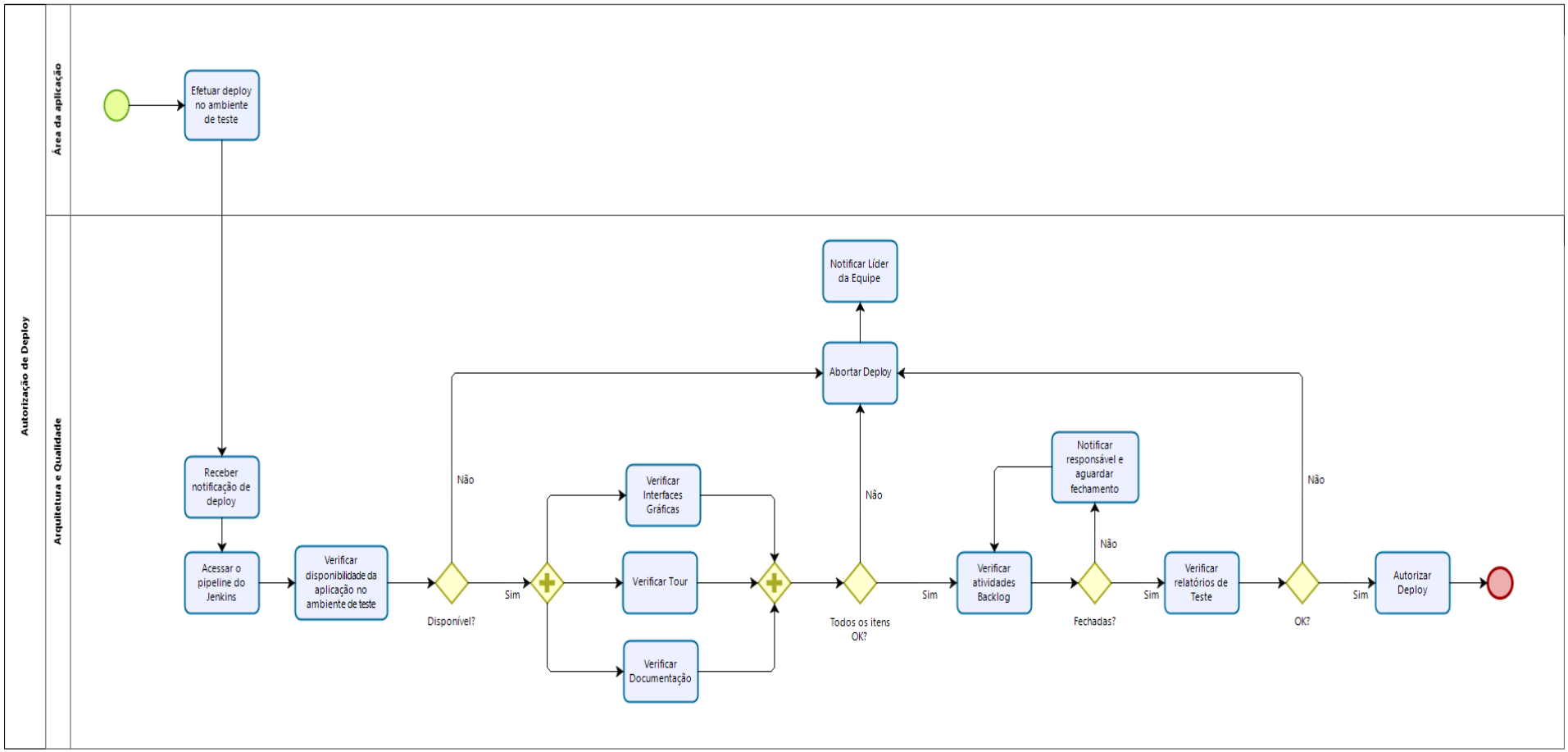

Diagrama 8 – Fluxo de Autorização de *Deploy*

8.4 Análise de qualidade de código

8.4.1 O Que:

A Ferramenta Jenkins aciona a Ferramenta Sonar para a execução da atividade de análise da qualidade do código-fonte.

8.4.2 Quem:

1. *Jenkins*.
2. Sonar.

8.4.3 Quando:

Após a validação da *build* e do *deploy*, descrita na Subseção 8.3.

8.4.4 Entradas:

Aplicação com *build* e *deploy* realizado.

8.4.5 *Saídas:*

Verificações de testes de qualidade de código realizado.

8.4.6 Fluxo:

Ir para a Atividade 8.5.

8.4.7 Tempo de espera:

Não se aplica

8.5 Análise de resultado de qualidade de código

8.5.1 O Que:

Arquiteto analista resultado de análise de qualidade de código.

8.5.2 Quem:

Arquiteto de Software.

8.5.3 Quando:

Após a execução da análise da qualidade do código-fonte, conforme descrita na Subseção 8.4.

8.5.4 Entradas:

Resultado da análise da qualidade do código-fonte realizada pela ferramenta Sonar.

8.5.5 Saídas:

Análise realizada.

8.5.6 Fluxo:

Se o código-fonte não for aprovado de acordo com as métricas de qualidade exigidas e configuradas na ferramenta, o processo deve ser abortado para efetuar as devidas correções. Se for aprovado, segue-se para a Atividade 8.6.

8.5.7 Tempo de Espera:

1 dia.

8.6 Gerar *Tag*

8.6.1 O Que:

Jenkins gera versão no repositório de fontes.

8.6.2 Quem:

Jenkins.

8.6.3 Entradas:

Código-fonte da aplicação a ser versionado.

8.6.4 Saídas:

Tag gerada.

8.6.5 Fluxo:

Segue para a Atividade 8.7.

8.7 Analisar escopo

8.7.1 O Que:

Arquiteto analisa escopo da entrega. Ou seja, se é uma versão somente de *bugfix* ou se contém alguma tarefa com Nova Funcionalidade ou Manutenção Evolutiva.

8.7.2 Quem:

Arquiteto de software.

8.7.3 Quando:

Após a geração da *Tag*, conforme descrito na Subseção 8.6.

8.7.4 Entradas:

Versão a ser entregue.

8.7.5 Saídas:

Análise realizada.

8.7.6 Fluxo:

Se a versão contiver manutenção evolutiva e/ou nova funcionalidade, o *Jenkins* envia um *email* para a equipe de Arquitetura e Qualidade e o fluxo deve ir a Atividade 8.9. Senão (somente *bugfix*), ir para a Atividade 8.8.

8.8 Verificar qualidade da entrega

8.8.1 O Que:

Analista de Qualidade verifica, dentre outros, se:

1. A aplicação passou nos testes automatizados?
2. Os artefatos da entrega estão em conformidade com o seu nível de criticidade exigidos para a Versão?
3. Se há entrega funcional e/ou de interface, está dentro dos padrões de UI estabelecidos?

Uma visão completa desta atividade pode ser obtida no Diagrama 8.

8.8.2 Quem:

Analista de qualidade.

8.8.3 Quando:

Após a análise do escopo da entrega, conforme descrito na Subseção 8.7.

8.8.4 Entradas:

Versão a ser analisada.

8.8.5 Saídas:

Análise realizada.

8.8.6 Fluxo:

1. Se a entrega/Versão **não** passar em todos os itens de conformidade relacionado a mesma, o *deploy* deve ser **abortado** e para cada não conformidade encontrada deve ser aberta uma tarefa de Manutenção Corretiva (com origem na Qualidade) para providências e correções. Um e-mail também deve ser enviado ao Líder da Equipe para ciência do mesmo.
2. Se a entrega/Versão **passar** em todos os itens de conformidade relacionado a mesma, ir para a Atividade 8.9.

Obs. [1]: Se houver alterações de banco de dados (SAC´s relacionados no [*release notes*]), o Analista de qualidade deverá solicitar via SAC ao Administrador de Dados a aplicação de tais alterações antes de aprovar o *deploy*.

Obs. [2]: Só haverão exceções para casos estritamente documentados (ex. Sistemas legados) ou com autorização da gerência (por *email* detalhado).

8.8.7 Tempo de Espera:

2 semanas.

8.9 *Build* de homologação e produção

8.9.1 O Que:

Jenkins realiza *build* de homologação e produção.

8.9.2 Quem:

Jenkins.

8.9.3 Quando:

Após a verificação da qualidade da entrega, conforme descrito na Subseção 8.8.

8.9.4 Entradas:

Versão a ser entregue.

8.9.5 Saídas:

Saída de *builds*.

8.9.6 Fluxo:

Ir para a Subseção 8.10.

8.10 Analisar *deploy* em Homologação

8.10.1 O Que:

Analista de Sistemas analisa se não houve nenhum problema no *deploy*.

8.10.2 Quem:

Analista de Sistemas.

8.10.3 Quando:

Após a criação da *build* em Homologação e Produção, conforme descrito na Subseção 8.9.

8.10.4 *Entradas:*

Build de homologação concluída.

8.10.5 *Saídas:*

Saída em console de *deploy*.

8.10.6 Fluxo:

1. Se o Analista de Sistemas abortar a *build* em algum momento, finaliza o processo.

2. Se o Analista de Sistemas autorizá-la, ir para a Atividade 8.11.

8.10.7 Tempo de Espera:

1 dia.

8.11 Realizar *deploy* em homologação.

8.11.1 O Que:

Jenkins realiza *deploy* em homologação.

8.11.2 Quem:

Jenkins.

8.11.3 Quando:

Após a análise do *deploy* em Homologação, conforme descrito na Subseção 8.10.

8.11.4 Entradas:

Versão a ser entregue.

8.11.5 Saídas:

1. Resultado do *deploy*.
2. Enviar *email* com [*release notes*] para o Escritório de Projetos.

8.11.6 Fluxo:

Ir para a Atividade 8.12.

8.12 Conferir *deploy*

8.12.1 O Que:

Analista de Sistemas confere se o *deploy* foi realizado com sucesso.

8.12.2 Quem:

Analista de Sistemas.

8.12.3 Quando:

Após a realização do *deploy* o ambiente de Homologação, conforme descrito na Subseção 8.11.

8.12.4 Entradas:

Comando de *deploy* no *Jenkins*.

8.12.5 *Saídas:*

Análise de *deploy*.

8.12.6 Fluxo:

1. Se o *deploy* foi realizado com sucesso, o Analista deve avisar o Líder da Equipe e o fluxo deve ir para a Atividade 8.13.
2. Se o *deploy* não foi realizado com sucesso, acionar o Arquiteto do Projeto e abortar processo.

8.13 Aguardar homologação

8.13.1 O Que:

Líder da Equipe informa o cliente (caso não seja *bugfix*) que a entrega foi realizada e está aguardando a homologação. Se for *bugfix*, o Líder da Equipe apenas informa o cliente sobre a entrega.

8.13.2 Quem:

Líder da Equipe.

8.13.3 Quando:

Após a conferência do *deploy* no ambiente de Homologação, conforme descrito na Subseção 8.12.

8.13.4 Entradas:

Jenkins aguardando homologação.

8.13.5 Saídas:

Não se aplica.

8.13.6 Fluxo:

1. Se a Versão for homologada, o Líder da Equipe deve autorizar o prosseguimento e o fluxo irá para a Atividade 8.14.
2. Se a Versão **não** for homologada, devidos a não conformidade(s), o cliente deverá solicitar (via SAC) a correção da(s) não conformidade(s). Para cada não conformidade deverá ser aberta uma tarefa no *Redmine*.

8.14 Efetuar deploy em produção

8.14.1 O Que:

Jenkins efetua o *deploy* no ambiente de produção e um e-mail é enviado para o Líder da Equipe, para o Gerente de Projetos e Sistemas e para o Cliente, informado da entrega.

8.14.2 Quem:

Jenkins.

8.14.3 Quando:

Após a homologação da versão e conforme definição do cliente.

8.14.4 Entradas:

Versão a ser implantada.

8.14.5 *Saídas:*

1. *Deploy* realizado.
2. *Email* enviado.

8.14.6 Fluxo:

Encerra o fluxo.

9 Anexo – Versionamento de SQL's

1. Criação da pasta Sql dentro do Repositório do Projeto no SVN/GIT, seguindo o padrão:

//url_do_repositório/grupo_sistema_projeto/sql

2. ~~Subpastas ?? (uma para o schema do BD (procedures, triggers), outra para as demandas de consultas/alterações associadas a Remines ou SACs)~~ (esse item nós não aprovamos, ficaria a critério do grupo, de acordo com a necessidade)
3. Padrão de nomes dos scripts:

Redm_6750-Correcao_Anulacao_Saldo_Negativo_201608.sql

Recomendações:

- retirar artigos, acentuações
- trocar espaços por "_" (underline)
- máximo 120 caracteres
- iniciar com número da demanda (importante para agrupar scripts da mesma demanda)
- no segundo token do nome utilizar verbo ou palavra que represente a natureza do script, como:
  - _consulta_xxx
  - _relatorio_xxx
  - _atualiza_xxx
  - _corrige_xxx
- terminar com .sql

4. Padrão de commit:

(refs #6750) Correção de Anulação de Saldo Negativo - folha 08/2016

Recomendações:

- Iniciar com expressão regular "(refs #número da demanda)" para que se possa construir a integração do Redmine com o SVN/GIT
- Copiar resumidamente a descrição da demanda, acrescentando no final algumas "tags" ou palavras_chave, no formato:

Palavras_chave: ~consignação ~corte_teto ~tce ~vinculos_ativos