ITRAC

Information Technology Research and Application Center

PROCESSO DE GESTÃO DE DEMANDAS DE MANUTENÇÃO DE SOFTWARE (GEDEM)

RELATÓRIO TÉCNICO

Brasília, Dezembro de 2017

UnB

MINISTÉRIO DA CIÊNCIA, TECNOLOGIA, INOVAÇÕES E COMUNICAÇÕES

BRASIL GOVERNO FEDERAL

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

ITRAC

P963 Processo de gestão de demandas de manutenção de software (GeDeM) : relatório técnico / Rejane Maria da Costa Figueiredo ... [et al.]. – Brasília : Universidade de Brasília, Campus Gama, 2017.
67 p. : il.

1. Manutenção de software. I. Figueiredo, Rejane Maria da Costa.

CDU 004

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

ITRAC

Processo de Gestão de Demandas de Manutenção de Software (GeDeM). Relatório técnico. FGA, UnB. Dezembro, 2017.

Autores:

Rejane Maria da Costa Figueiredo (FGA/UnB, ITRAC)

Elaine Venson (FGA/UnB, ITRAC)

Augusto Samuel Modesto Clementino (ITRAC)

Ana Paula Vargas de Noronha (FGA/UnB, ITRAC)

Jads Victor Paiva dos Santos

Pesquisa realizada com financiamento do Ministério das Comunicações, Projeto de Cooperação "Framework de Soluções de Tecnologia da Informação para o MC".

ÍNDICE

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

ITRAC

LISTA DE FIGURAS

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

ITRAC

LISTA DE TABELAS

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

ITRAC

1 INTRODUÇÃO

Este Relatório é resultante de uma das frentes de pesquisa e desenvolvimento do **Projeto P&D-MC/UnB** (Projeto de Pesquisa e Desenvolvimento entre a Universidade de Brasília – UnB, Faculdade FGA e o Ministério das Comunicações - MC), oriundo de um termo de cooperação entre a UnB e o Ministério. Uma das metas do Projeto foi atender a demanda do Ministério quanto a definição de um processo de que possibilitasse gerir as demandas de manutenção de software pela terceirizada, no caso, fábricas de software e consultorias em gestão da qualidade, empregando valores e princípios das metodologias ágeis.

Inicialmente foi necessário inventariar os softwares legados do Ministério. Para isso, foram definidos atributos de inventariação dos softwares. Em seguida, foi definido o Processo de *Gestão de Demandas de Manutenção de Software (GeDeM)*. O projeto compreendeu a definição, avaliação e implantação deste processo no Ministério.

Como produção técnica, o processo foi definido, implantado, e validado no MC.

Como produção acadêmica, até o momento, foram geradas algumas publicações em conferências internacionais, tais como:

- Noronha, A. P. V.; Venson, E.; Figueiredo, R. M. C.; Modesto, A. S. C. Applying Kanban to Manage Outsourced Maintenance Services: An Action Research in a Brazilian Government Agency. In: CIbSE Conference IberoAmerican on Software Engineering (ESELAW Experimental Software Engineering Track), 22-23 May, 2017, Buenos Aires, Argentina.
  Link: Indisponível

- Santos, Jads Victor Paiva dos; Figueiredo, R. M. C; Noronha, Ana Paula Vargas de; Venson, Elaine. Using Kanban in Outsourced Government Projects of Management Maintenance Demands: a Descriptive Research. In: 13th CONTECSI International Conference on Information Systems and Technology Management, 2016. p. 4147
  Link: http://www.contecsi.fea.usp.br/envio/index.php/contecsi/13CONTECSI/paper/view/4204

Além de outras produções relacionadas a esta temática, apresentadas no ANEXO 1.

- Coletânea de artigos publicados em conferências, nacionais e internacionais, relacionados ao processo de desenvolvimento de software, de inventariação de sistemas, e de transferência de conhecimento.

- Coletânea dos Trabalhos de Conclusão de Curso da Faculdade GAMA – FGA, relacionados ao tema deste Relatório.

Em 2016, houve a fusão do Ministério das Comunicações com o Ministério da Ciência, Tecnologia e Inovação, surgindo o Ministério da Ciência, Tecnologia, Inovação e Comunicações – MCTIC. O **Projeto P&D-MC/UnB** foi vinculado a esse novo Ministério.

Neste relatório, apresenta-se o Processo GeDeM.

1.1 Contexto

Sistemas, software, ou aplicações de software devem ser constantemente adaptados para atender às necessidades de seus usuários. Mesmo após a sua entrega, qualquer software deve ser mantido continuamente para não se tornar obsoleto (PRESSMAN, 2006).

O processo caracterizado pela realização de modificações em um sistema/software, depois que ele foi colocado em uso, é chamado de Manutenção de Software (ISO/IEC, 2006) (BENNETT; XU, 2003). De acordo com a ISO/IEC 14765 (2006), ele deve conter as atividades e tarefas necessárias para modificar um produto de software existente, preservando sua integridade.

O processo de manutenção de software deve ser definido e compreendido pelos envolvidos, ou seja, é necessário que suas etapas sejam estabelecidas e difundidas em toda a organização desde a identificação de uma demanda até a sua resolução e liberação para o cliente (APRIL, 2010). Para propor esse tipo de processo, devem ser considerados quatro importantes fatores que influenciam diretamente na manutenção de software: o time de desenvolvimento, as ações de gerenciamento, o código e o usuário final (BHATT; SHIROFF; MISRA, 2004).

O time de desenvolvimento deve estar motivado e não focar apenas no aspecto técnico da manutenção de software, se concentrando nas necessidades de negócio do cliente. As atitudes envolvidas no gerenciamento devem considerar o ambiente instável, onde a pressão para que ocorra a solução de demandas urgentes é usual. A qualidade do código influencia diretamente no esforço e eficiência da manutenção realizada. Finalmente, o usuário final deve estar diretamente envolvido no projeto de manutenção, para que o trabalho realizado corresponda integralmente ao solicitado (BHATT; SHIROFF; MISRA, 2004).

Considerando os fatores citados acima e levando em conta que cada um deles apresenta características e perfis distintos, um processo de manutenção de software adequado necessita se adaptar a essas particularidades. Ele deve possuir resistência mínima a mudanças e, ao mesmo tempo, buscar a produtividade e a satisfação do cliente.

Um *framework* que apresenta esse conjunto de singularidades e permite a otimização de processos em diferentes contextos é o *Kanban* (OHNO, 1997). Ele surgiu no final dos anos 40 com o intuito de controlar o Sistema Toyota de Produção, limitando a manufatura a partir da demanda e determinando que a realização de um novo trabalho ficasse condicionada à disponibilidade para a sua execução (ANDERSON, 2010).

O *Kanban* começou a ter grande adesão na indústria de software em 2007, depois da divulgação dos resultados obtidos em desenvolvimento de aplicações em duas grandes conferências do ramo. A possibilidade de uma equipe visualizar o trabalho em progresso, se auto-organizar e, ainda, moldar seu fluxo de trabalho de acordo com um ambiente específico, despertou o interesse da comunidade (ANDERSON, 2010).

Tendo em vista que a adoção da abordagem *Kanban* nas áreas de manutenção e operação de sistemas provou trazer bons resultados (BOEG, 2011), um grupo de alunos e professores da Universidade de Brasília (UnB) propôs um processo de gestão de demandas de manutenção para um órgão público, utilizando essa metodologia.

O órgão, Ministério das Comunicações, recorre à terceirização de serviços de TI, prática que tem se tornado comum nas empresas (ALARANTA; JARVENPAA, 2010) e na Administração Pública Federal. Medidas que dizem respeito às diretrizes para essas contratações vêm sendo implementadas pelo Governo Federal (CRUZ; ANDRADE; FIGUEIREDO, 2011), dentre elas a edição da Instrução Normativa 04 e do Guia Prático para Contratação de Soluções de TI (BRASIL, 2014a).

Neste contexto e, com a aprovação do processo proposto pelo grupo da UnB, iniciou-se a avaliação para comprovar a sua eficiência, o que caracteriza o foco principal deste trabalho. Essa avaliação foi precedida de um diagnóstico para identificar os problemas do processo vigente, e de um planejamento visando à definição do projeto-piloto, dos participantes e da ferramenta *Kanban* a ser utilizada. Com o estabelecimento de todas as tarefas preparatórias para a execução contínua do processo, os testes começaram a ser realizados em três ciclos, possibilitando a aplicação de refinamentos.

Portanto, o presente trabalho pretende avaliar um processo de gestão de demandas de manutenção de software, empregando o *Kanban* na área de TI do Ministério X, com a finalidade de otimizar este serviço.

Este trabalho está organizado em cinco capítulos. Neste Capítulo 1 – Introdução, são apresentados o contexto, o problema e o objetivo da pesquisa.

No Capítulo 2 – Contratações de Serviços de Tecnologia da Informação por Organizações Públicas Brasileiras, são apresentados conceitos e características do processo da contratação de serviços de TI por órgão do governo.

O Capítulo 3 – Manutenção de Software, contém definições, categorias e atividades relativas ao processo de manutenção de software, além de destacar a importância da realização desse serviço e os diversos problemas que o envolvem.

O Capítulo 4 – *Kanban*, dispõe sobre a origem, os conceitos, o funcionamento e os princípios fundamentais que norteiam a metodologia *Kanban*, apresentando, ainda, o relato de dois estudos de caso nos quais a utilização da abordagem resultou em diversas vantagens.

No Capítulo 5 – Processo de Gestão de Demandas de Manutenção de Software, é apresentado o processo definido, bem como o detalhamento de suas atividades, papeis e artefatos.

No Capítulo 5 – Apresentam-se as considerações finais deste trabalho.

Anexo 1 – Produção Acadêmica - apresenta-se uma coletânea de artigos publicados em conferências e *Trabalhos de Conclusão de Curso* relacionados.

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

2 CONTRATAÇÕES DE SERVIÇOS DE TECNOLOGIA DA INFORMAÇÃO POR ORGANIZAÇÕES PÚBLICAS BRASILEIRAS

Este capítulo trata da importância do processo de contratação de serviços de TI pela Administração Pública Federal, os órgãos envolvidos e as fases que o compõem, abordando modelos, normativos e a legislação pertinente.

2.1 Importância na Contratação de TI pelo Setor Público

No Brasil, a contratação de serviços de desenvolvimento de software pela Administração Pública Federal vem ocorrendo cada vez mais, contribuindo para o fortalecimento e crescimento dessa atividade. O Decreto-Lei Nº 200 (BRASIL, 1967) estabelece que a administração deve recorrer, mediante contrato e sempre que possível, à execução indireta dos serviços que apoiem a sua área fim, atividade comumente conhecida como "terceirização de serviços".

A contratação de TI tem se tornado uma prática comum nas empresas, podendo trazer algumas vantagens e desvantagens. A maior dependência do fornecedor, por exemplo, pode limitar o poder do cliente, colocando em risco sua flexibilidade estratégica, além de gerar elevação dos custos e redução da qualidade do serviço (ALARANTA; JARVENPAA, 2010). Por outro lado, o fornecedor tem a oportunidade de focar nas atividades empresariais para impulsionar o negócio e gerar valor para organizações (CRUZ; ANDRADE; FIGUEIREDO, 2011).

No âmbito da Administração Pública, a licitação, regida pela Lei nº 8.666, de 21 de junho de 1993 (BRASIL, 1993), é o processo formal utilizado para selecionar, através de edital, a proposta mais vantajosa que atenda suas necessidades de bens e serviços (BRASIL, 2010).

2.2 Sistema de Administração dos Recursos de Tecnologia da Informação (SISP)

O sistema responsável pela gestão dos serviços de TI, na esfera federal, é o Sistema de Administração dos Recursos de Tecnologia da Informação (SISP), criado em 1994 e atualizado em 2011, para organizar a operação, controle, supervisão e coordenação dos recursos de informação e informática dos órgãos públicos federais. Ele é responsável por promover a integração e a articulação entre os programas de governo, projetos e atividades, bem como por definir políticas, diretrizes e normas para a gestão dos recursos de TI (BRASIL, 2014b).

O SISP integra mais de 210 órgãos do Executivo Federal, sendo a Secretaria de Logística e Tecnologia da Informação (SLTI) sua unidade central. Um dos principais objetivos desta Secretaria é normatizar, promover e coordenar ações junto aos órgãos do SISP quanto à gestão e governança de tecnologia da informação, gestão de pessoas e capacitação, e melhoria de processos de desenvolvimento de sistemas (BRASIL, 2012a).

Em 2008, a SLTI elaborou a Instrução Normativa 04 - IN 04/2008 (BRASIL, 2008), atualizada em 2014 para Instrução Normativa 04/2014 - IN 04/2014 (BRASIL, 2014c), que disciplina as contratações de Soluções de TI pelos órgãos e entidades integrantes do SISP.

A partir deste normativo, as contratações de TI passaram a ser cada vez mais vinculadas ao Planejamento Estratégico Institucional dos órgãos do Governo Federal. Além disso, elas devem estar alinhadas com o Plano Diretor de Tecnologia da Informação (PDTI), que é o instrumento de diagnóstico, planejamento e gestão dos recursos e processos de TI para atender às demandas de um órgão ou entidade em um determinado período.

2.3 Instrução Normativa Nº 4 e o Modelo de Contratação de Soluções de TI

A atual Instrução Normativa Nº 4 (BRASIL, 2014c) consolidou a revisão de um conjunto de boas práticas para contratação de Soluções de TI pelos órgãos integrantes do SISP do Poder Executivo Federal (BRASIL, 2014a). Ela estabelece que o processo de contratação de soluções de TI deve ser realizado nas três fases a seguir especificadas, definindo seus respectivos papéis e artefatos:

- **Planejamento da Contratação**: visa identificar a demanda da contratação, levando em conta os objetivos estratégicos e as necessidades corporativas da instituição, assim como seu alinhamento com o PDTI;
- **Seleção do Fornecedor**: visa analisar as sugestões feitas pela área de licitações e jurídica quando do recebimento do termo de referência ou projeto básico, cabendo à área de licitações conduzir todas as etapas dessa fase;
- **Gerenciamento do Contrato**: visa acompanhar e garantir a adequada prestação do serviço e o fornecimento dos bens que compõem a solução de TI durante todo o período de execução do contrato;

A IN 4 contém três capítulos, sendo contempladas no capítulo dois as três fases do processo de contratação de TI, conforme a Figura 1.

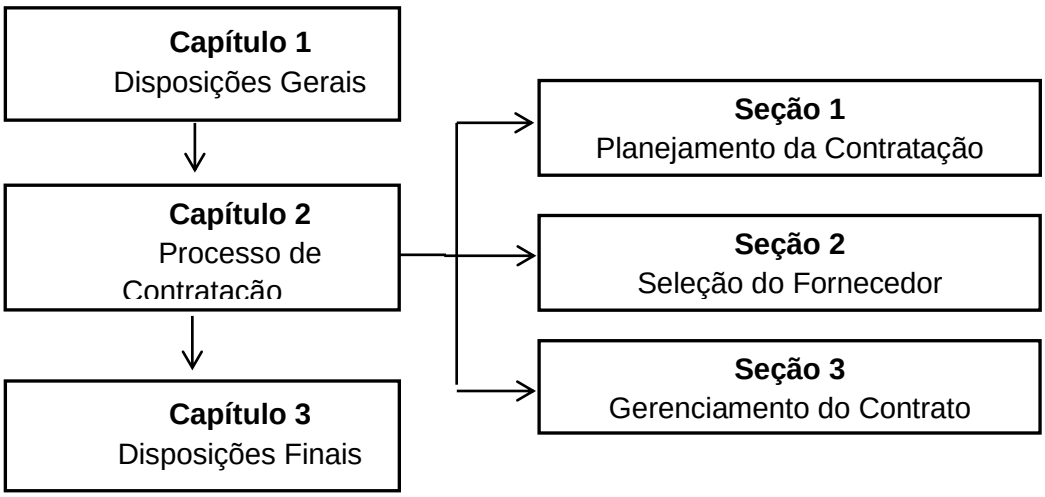

Figura 1 - Estrutura da IN 4/2014. Fonte: (BRASIL, 2014a)

Como produto do processo de revisão da IN 04/2008, atualizado pela IN 04/2014, foi elaborado o Guia Prático para Contratação de Soluções de TI (BRASIL, 2014a), denominado Modelo de Contratação de Soluções de TI (MCTI). Neste modelo os processos, atividades, artefatos e atores de cada fase descrita pela norma são detalhados com vistas a apoiar os profissionais na concretização das contratações de serviços de TI.

A Figura 2 apresenta as três fases do processo e os respectivos artefatos que irão implicar no início e finalização de cada uma delas. A fase de Planejamento da Contratação só inicia com a elaboração do Documento de Oficialização da Demanda e possui quatro etapas: Instituição da Equipe de Planejamento da Contratação, Estudo Técnico Preliminar da Contratação, Análise de Riscos e Geração do Termo de Referência ou Projeto Básico, tendo esta última a obrigação de consolidar todas as principais informações obtidas anteriormente.

Após a aprovação do Termo de Referência, segue-se a fase de Seleção do Fornecedor, cuja responsabilidade é da Área de Licitações do órgão ou entidade que está conduzindo o processo de contratação, cabendo à Área de Tecnologia da Informação o apoio a alguns processos. O final desta fase ocorre com a elaboração do Contrato.

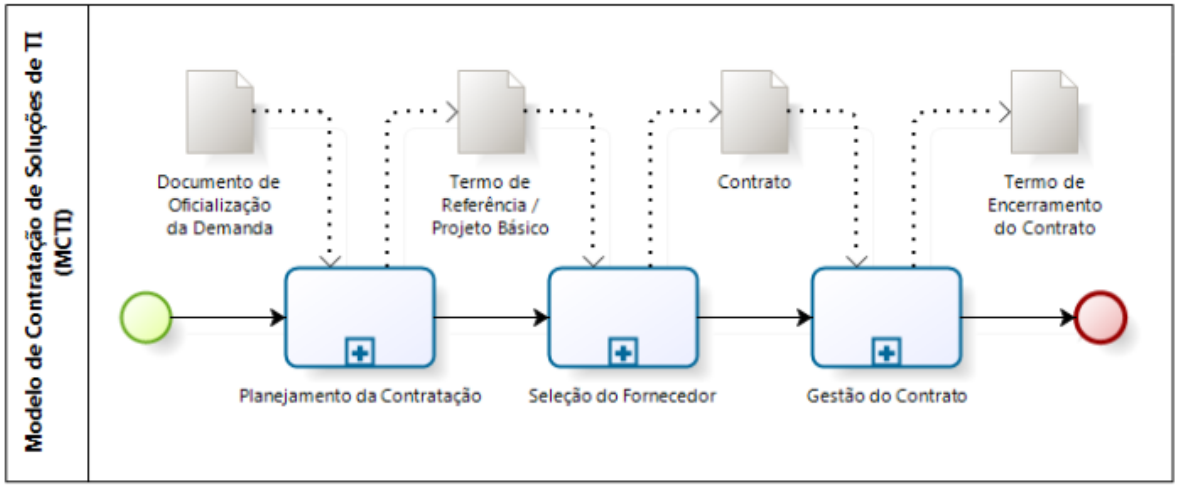

Figura 2 - Modelo de Contratação de TI. Fonte: (BRASIL, 2014a)

A Gestão do Contrato, última fase do MCTI, visa o acompanhamento e a garantia da adequada prestação dos serviços e do fornecimento de bens que compõem a Solução de TI durante todo o período de execução do contrato. O processo de encerramento do Contrato é formalizado pela assinatura do artefato Termo de Encerramento do Contrato, pelos representantes legais da Contratante e da Contratada.

2.4 Outros Modelos de Contratação

Além da IN 4 e do MCTI, existem dois outros documentos sobre o tema que podem ser utilizados para apoiar os gestores de contratos no planejamento das contratações: o Processo de Contratação de Serviços de TI – PCSTI (CRUZ; ANDRADE; FIGUEIREDO, 2011) e o Guia de Boas Práticas, elaborado pelo Tribunal de Contas da União – TCU, (BRASIL, 2012b).

O PCSTI define o processo em termos de atores, atividades e tarefas, apresentando inclusive *templates* de seus artefatos propostos. Sua finalidade é dar suporte à contratação de serviços de TI que satisfaçam as necessidades de negócio da organização pública contratante, alinhada à sua estratégia e à legislação brasileira, seguindo os princípios de eficácia, efetividade, economicidade, legalidade e legitimidade dos projetos (CRUZ; ANDRADE; FIGUEIREDO, 2011).

Em 2012, O TCU lançou o Guia de Boas Práticas em contratação de soluções de Tecnologia da Informação (BRASIL, 2012b) com o objetivo de ajudar os gestores públicos a planejar as contratações de TI e evitar problemas já detectados. O guia apresenta recomendações referentes ao planejamento das contratações, sob o ponto de vista do controle externo da Administração Pública Federal, baseadas na legislação, na jurisprudência e nas melhores práticas do mercado, incluindo sugestões de controles internos para tratar riscos relativos ao processo de contratação de soluções de TI.

3 MANUTENÇÃO DE SOFTWARE

Este capítulo apresenta conceitos relativos à manutenção de software, incluindo sua categorização, atividades decorrentes da sua implantação e os desafios que ela impõe. Nele é destacada a importância de que esta etapa seja bem planejada e executada de modo a atender as demandas dos usuários.

3.1 Definições

Manutenção de Software, segundo a norma ISO/IEC 14764 (2006), é um processo centrado na modificação de um sistema após sua entrega ao cliente. Ela ocorre porque problemas ou necessidades são identificados quando o software é colocado em operação.

De acordo com Grubb e Takang (2003), existem diversas definições de Manutenção de Software na literatura, algumas mais abrangentes e outras específicas. As encontradas com maior frequência estão associadas à correção de erros e à necessidade de adaptar sistemas quando o ambiente operacional ou requisitos originais são alterados.

Para Hunt, Turner e McRitchie (2008), a Manutenção de Software inclui todas as atividades realizadas sobre o sistema após sua liberação, que devem ser distinguidas das modificações executadas na etapa de Desenvolvimento. April e Abran (2008) também identificam essas duas fases do ciclo de vida do software, ressaltando que, diferentemente do estágio de Desenvolvimento, o de Manutenção está estruturado para atender desafios, como solicitações inesperadas e urgentes de usuários e acompanhamento contínuo do software.

Outro meio utilizado por Hunt, Turner e McRitchie (2008) para definir a Manutenção de Software é sua comparação com a Manutenção de Hardware. Segundo eles, softwares não se desgastam fisicamente, mas, com o tempo, entram em desuso e podem ser entregues com falhas não descobertas na etapa de Desenvolvimento.

Apesar dos conceitos de Manutenção de Software destacados, April (2010) afirma que é comum associar a execução dessa etapa a uma mera atividade para reparo de erros em sistemas. A falta de compreensão de todas as ações que envolvem esse processo leva, muitas vezes, a uma percepção de que organizações de manutenção de software são caras e ineficientes. Porém, estudos mostram que grande parte do esforço despendido nesta etapa do ciclo de vida do software acrescenta valor para as instituições.

3.2 Categorização

A Manutenção de Software pode ser classificada de diversas formas. Uma delas foi estabelecida por Tripathy e Naik (2008) e se baseia na intenção, isto é, no objetivo do

desenvolvedor ao realizar tarefas específicas de manutenção no sistema. Inicialmente, ela foi desenvolvida por E. Burton Swanson (1976) que a dividiu em três tipos:

- Manutenção Corretiva: Tem como objetivo o reparo de defeitos nas aplicações, isto é, a correção de problemas que surgem durante a utilização do sistema. Essas falhas podem ser de processamento, comumente atribuídas a *bugs* de software, de performance ou até mesmo de implementação, como, por exemplo, violações em padrões de programação.
- Manutenção Adaptativa: É executada em resposta a modificações no ambiente externo, visando a adequação do software ao contexto no qual ele deve operar. Mudanças no hardware ou a necessidade de instalação de uma nova versão de um sistema operacional, por exemplo, podem requerer este tipo de manutenção.
- Manutenção Perfectiva: Visa à realização de modificações apenas no sentido de melhorar o software, como por exemplo a inclusão de novas funcionalidades, a eliminação de ineficiências no processamento, um aumento no desempenho, ou, até mesmo, um aprimoramento na própria manutenibilidade do sistema.

Posteriormente, essas três definições foram incorporadas pela norma ISO/IEC 14764 (2006), tendo sido introduzida uma quarta categoria denominada Manutenção Preventiva. Ela é realizada a partir de modificações após a liberação do software com o intuito de detectar e corrigir falhas latentes, antes que se tornem falhas operacionais.

Mesmo caracterizados individualmente, os quatro tipos de manutenções podem se relacionar de várias maneiras. Por exemplo, ao modificar uma aplicação através da introdução de um novo sistema operacional (manutenção adaptativa), há uma chance de novos *bugs* serem inseridos, surgindo a necessidade de identificá-los e tratá-los (manutenção corretiva). De forma semelhante, adicionar um algoritmo no código, visando maior eficiência (manutenção perfectiva), pode demandar sua restruturação de modo a evitar problemas futuros (manutenção preventiva) (GRUBB; TAKANG, 2003).

3.3 Processo de Manutenção de Software

Para uma organização ser considerada madura é necessário que ela tenha instituído o seu processo de manutenção de software (APRIL, 2010). Normas internacionais com o intuito de estabelecer atividades referentes a esse serviço começaram a surgir em 1970 e, em geral, apresentavam três etapas: Compreensão do Problema, Implementação da Modificação e

Validação da Mudança. Apesar da existência desses modelos, muitas organizações ainda não apresentam este processo bem definido (APRIL; ABRAN, 2008).

Segundo Yongchang et al. (2011), o processo é um meio essencial para assegurar a preservação da qualidade, pois ele auxilia na detecção precoce de problemas. A Figura 3 apresenta o processo de manutenção de software identificado por ele:

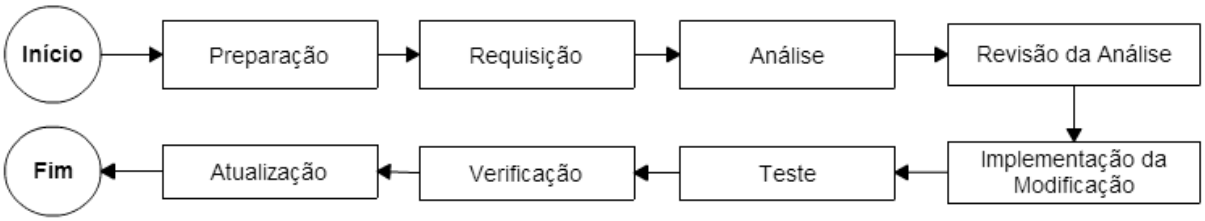

Figura 3 - Processo de Manutenção de Software. Fonte: (YONGCHANG, 2011)

A etapa de "Preparação" inclui a designação da equipe, o estabelecimento dos canais de comunicação, a identificação dos treinamentos necessários e a elaboração e aprovação do Plano de Manutenção de Software. O estágio seguinte é denominado "Requisição" e tem início a partir de um pedido de mudança no sistema, geralmente efetuado pelo usuário. A "Análise", terceira fase do processo, compreende o entendimento do problema, constituindo-se basicamente no estudo dos aspectos e impactos que envolvem a requisição recebida. Na "Revisão da análise", avalia-se o grau de dificuldade da implementação da demanda, incluindo a sua capacidade de resolução e, dependendo dos resultados obtidos, são definidos os métodos de execução.

Após a realização completa da análise, ocorre a quinta etapa: Implementação da Modificação. Nela são feitas as alterações no código e, ao final, é gerado um relatório contendo as informações sobre a modificação efetuada. Em seguida, são inseridos os testes e há uma validação do resultado, garantindo a qualidade na manutenção de software executada ("Teste" e "Verificação"). Para concluir o processo, o último estágio é a "Atualização", que compreende a publicação do software e sua disponibilização ao usuário.

A norma ISO/IEC 14764 (2006) define um *framework* cujo objetivo é orientar o planejamento e a execução de um processo de manutenção de software, independentemente do tamanho, complexidade, criticidade ou domínio de aplicação do sistema. Ela é um guia internacional que contém a descrição de atividades e tarefas necessárias para modificar um produto de software, preservando a sua integridade. Cabe ressaltar que, para executar as atividades descritas no *framework*, é essencial que os responsáveis pela manutenção assegurem que o processo definido previamente está sendo seguido.

O processo estabelecido pela ISO/IEC 14764 (2006) pode ser aplicado a qualquer modelo de ciclo de vida de desenvolvimento de software. Ele suporta a manutenção desde o seu início, com a elaboração do plano a ser cumprido, até o seu fim, quando a aplicação entra em desuso. As atividades que o compõem são:

1. **Implementação do Processo**: São estabelecidos os planos e procedimentos a serem executados na fase de manutenção, além da definição das práticas de recebimento, registro e acompanhamento de requisições de mudança.
2. **Análise do Problema e da Modificação:** As requisições de mudança são analisadas com o intuito de verificar seus possíveis impactos na organização, no sistema em que a solicitação foi realizada e nas aplicações que apresentam interface com ele. Aspectos como o tipo, o escopo e a criticidade da requisição devem ser estipulados. Após a análise, alternativas de implementação das modificações devem ser identificadas, documentadas e aprovadas.
3. **Implementação da Modificação:** As mudanças aprovadas na atividade anterior são implementadas e testadas.
4. **Revisão e Aceitação da Modificação:** Visa garantir que as modificações foram implementadas corretamente, de acordo com o que foi planejado e especificado. Revisões são efetuadas para verificar a integridade do sistema e, se o resultado obtido for satisfatório, a mudança é aprovada.
5. **Migração:** Levando em conta a necessidade de o sistema funcionar em diferentes ambientes, ações necessárias para migrá-lo devem ser planejadas, aprovadas e executadas.
6. **Retirada:** Se o sistema chegou ao fim de sua vida útil, uma análise é feita para que ele seja retirado de forma segura. Um plano de descontinuidade é elaborado, os interessados são notificados e os registros relacionados são arquivados.

A Figura 4 mostra uma visão geral do processo e como suas atividades interagem.

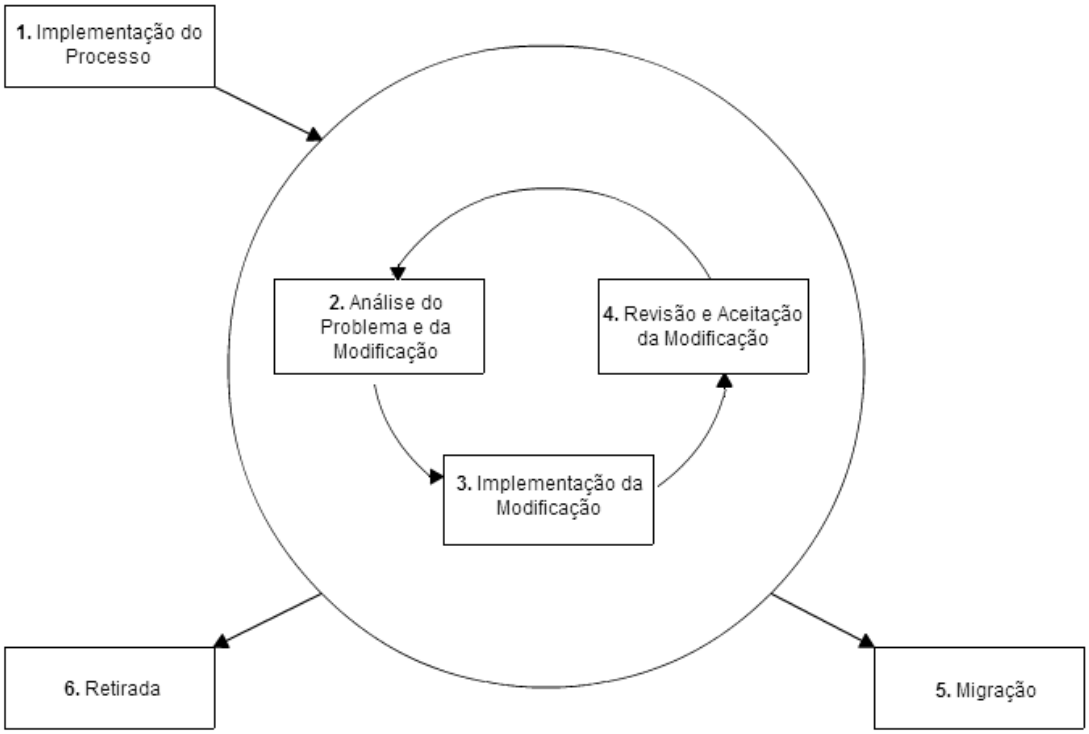

Figura 4 - Processo de Manutenção de Software da ISO/IEC 14764. Fonte: (ISO/IEC 14764, 2006, adaptado)

Enquanto as atividades 1, 5 e 6 não costumam ser executadas mais de vez no processo de manutenção, as ações 2, 3 e 4 se comportam de forma cíclica e somente são finalizadas quando a modificação está em conformidade e é, de fato, aprovada. De qualquer forma, é importante ressaltar que organizações podem adaptar o modelo e suas atividades de acordo com o contexto em que estão inseridas (APRIL; ABRAN, 2008).

Diversas atividades de manutenção de software ainda não foram formalizadas e, consequentemente, não são tratadas em normas, mas são observadas diariamente na indústria (APRIL; ABRAN, 2008).

Modelos que estabelecem o processo de manutenção de software apresentam diferentes focos, sendo que alguns se atentam mais a questões econômicas, outros ao produto e outros ao próprio processo. Todos possuem vantagens e desvantagens, não existindo um modelo único e aplicável às diversas situações. A melhor solução, muitas vezes, é utilizá-los em conjunto para obter o resultado que agregue mais benefícios para as organizações (YONGCHANG et al., 2011).

3.4 Problemas

Um dos grandes desafios enfrentados por engenheiros de software é gerenciar mudanças que ocorrem em todo o ciclo de vida de um sistema (HUNT; TURNER; MCRITCHIE, 2008). Quando se trata da fase de manutenção, essa tarefa se torna ainda mais difícil, pois os softwares podem ter sido desenvolvidos há muitos anos, com linguagens e processos agora considerados ineficientes, e para computadores com severas limitações. Além disso, equipes de manutenção devem possuir um conhecimento amplo sobre os sistemas mantidos, para que realizem alterações adequadas, seguras e rápidas, em conformidade com os requisitos. (HAVLICE et al., 2009).

Do ponto de vista do usuário e do desenvolvedor, a Manutenção de software muitas vezes é identificada como um problema (REN et al., 2011). Clientes costumam considerá-la uma fase cara e demorada, sobretudo pela falta de compreensão das atividades que constituem o processo, não enxergando o real valor que ela envolve (APRIL, 2010). Segundo April e Abran (2008), uma melhor comunicação entre a equipe de manutenção e o cliente certamente auxiliaria na mudança dessa perspectiva.

A equipe de manutenção que frequentemente lida com sistemas implementados por outras pessoas deve se familiarizar rapidamente com o código, sem que o serviço seja interrompido. Ela recebe solicitações que precisam ser resolvidas com urgência, vivenciando um contexto de pressão, que provoca o sentimento de pouco ou nenhum controle sobre a qualidade do código alterado. Para piorar, as atividades realizadas pela equipe são comumente acusadas de degradar a estrutura interna de aplicações e dificultar as próximas manutenções requeridas (APRIL; ABRAN, 2008).

Para que os usuários e desenvolvedores não classifiquem o processo de manutenção de software como um transtorno e comecem a valorizá-lo, ele deve ser executado de forma planejada, através de acordos prévios e divulgação de estimativas de gastos (REN et al., 2011). O custo elevado, dentre os diversos problemas que envolvem o processo de manutenção de software, ocupa posição de destaque, pois é afetado significativamente por todos os demais (PFLEEGER, 2004).

3.5 Custo

A manutenção consome uma parcela significativa dos recursos financeiros no ciclo de vida de um software (ISO/IEC, 2006) (GRUBB; TAKANG, 2003) e, segundo estudos realizados nas últimas décadas, a tendência é de que esse gasto aumente cada vez mais (RASHID; WANG; DORNER, 2009).

Engenheiros de software acreditavam que, com a utilização de novos processos e linguagens de programação, ocorreria uma redução radical nos custos associados à

manutenção. Porém, os sistemas continuam a ser entregues com um número excessivo de defeitos e, consequentemente, os gastos permanecem altos (HUNT; TURNER; MCRITCHIE, 2008).

Enquanto nos anos 70 a etapa de desenvolvimento se destacava por apresentar um custo superior ao de todas as outras, no ano 2000 as despesas de manutenção correspondiam até 80% do orçamento referente ao ciclo de vida de um sistema (PFLEEGER, 2004). De acordo com Yongchang et al. (2011), se tratando de uma manutenção em larga escala, os gastos são em média quatro vezes maior do que os exigidos no estágio de desenvolvimento do software.

A Figura 5 (GRUBB; TAKANG, 2003) apresenta o custo relativo à correção de erros ao longo do ciclo de vida de um sistema, indicando que, quanto mais tarde um defeito é detectado, maiores são as despesas para repará-lo.

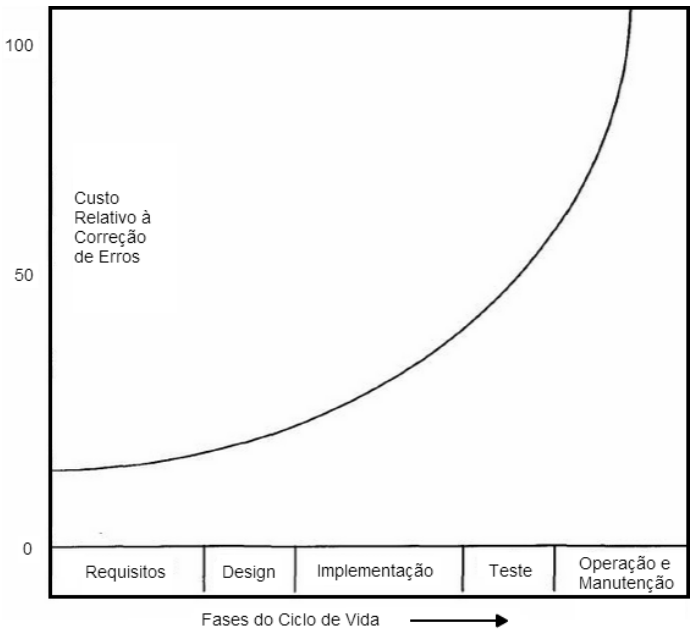

Figura 5 - Custo Relativo à Correção de Erros no Ciclo de Vida do Software. Fonte: (GRUBB; TAKANG, 2003)

Segundo Ren et al. (2011), são vários os fatores que afetam o custo da manutenção, podendo ser reunidos em dois grupos: aspectos não técnicos e aspectos técnicos. O primeiro deles é influenciado pela:

- Experiência da equipe: Quando os responsáveis pela manutenção já conhecem o sistema, maior é a facilidade para realizar as modificações demandadas.
- Rotatividade e disponibilidade da equipe: entender o software a ser mantido requer tempo e, todas as vezes que integrantes envolvidos no processo de manutenção forem substituídos, a tarefa de aprendizagem deverá ser repetida.
- Tempo de vida da aplicação: À medida que os softwares se tornam antigos e não há manutenção adequada, mais difícil será compreendê-los e modificá-los.
- Ambiente Externo: Mudanças nas regras de negócio ou em qualquer aspecto externo ao produto de software implicam evoluções no sistema.
- Ambiente de Suporte: Se o software não estiver preparado para eventuais adaptações como, por exemplo, uma modificação no hardware que o sustenta, o esforço para alterá-lo será elevado.
- Necessidades do Usuário: Quando o sistema está em operação o cliente consegue entender de maneira profunda suas particularidades, possibilitando a identificação de necessidades e o surgimento de novos pedidos de mudanças.

O outro grupo constatado representa os impactos causados por aspectos técnicos, incluindo:

- Complexidade do Software: Se há complexidade na estrutura da aplicação é preciso mais tempo para entendê-la e modificá-la, aumentando consideravelmente a carga de trabalho da equipe.
- Qualidade da Documentação: Quando não há documentação do sistema ou quando ela é insuficiente, localizar e solucionar um problema se torna uma tarefa árdua.
- Tecnologia de Gerência de Configuração: Monitorar e notificar mudanças de forma eficiente em uma ferramenta de gerenciamento de configuração pode contribuir para o maior controle de custos de manutenção.
- Boas Práticas de Programação: Adotar princípios e métodos de engenharia de software que visem, por exemplo, melhorar a coesão e reduzir o acoplamento, auxilia a implementação de mudanças.

- Tamanho do Banco de Dados: Quanto maior o banco, maior é o esforço para reorganizá-lo ou reconstruí-lo e, por conseguinte, para mantê-lo.

Muitas das dificuldades enfrentadas por equipes de manutenção decorrem de um processo falho de desenvolvimento, que transferiu problemas não contornados anteriormente. Por isto, é fundamental que as decisões de implementação e de gerenciamento feitas durante o processo de desenvolvimento sejam analisadas, possibilitando que os gastos de manutenção resultantes não sejam afetados significativamente (HUNT; TURNER; MCRITCHIE, 2008).

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

ITRAC

4 Metodologia *KANBAN*

Neste Capítulo são apresentados conceitos relativos à metodologia *Kanban*, incluindo sua origem, algumas de suas definições encontradas na literatura, seu funcionamento e princípios fundamentais. Além disso, são descritos dois estudos de caso onde a aplicação da abordagem resultou em diversos benefícios.

4.1 Origem

Kanban é uma palavra de origem japonesa que significa "cartão" ou "sinalização". No final dos anos 40, este termo deu nome a um mecanismo concebido pela empresa Toyota, para controlar sua produção (ANDERSON, 2010). Taiichi Ohno, um dos responsáveis pela criação do Sistema Toyota de Produção, desenvolveu a metodologia *Kanban*, com o intuito de reduzir custos e gerenciar a utilização de máquinas (GROSS et al., 2003).

Uma das filosofias de manufatura que surgiu no mesmo período de criação do *Kanban*, dando suporte a esse mecanismo, é o *Just in Time* (JIT). Seu objetivo é produzir o que é necessário, na quantidade especificada, no momento requisitado (ANDERSON, 2010) (MAREK; ELKINS; SMITH, 2001). Esta abordagem indica que a produção não deve ser iniciada antecipadamente à demanda, evitando estoques parados, desperdício e retrabalho. Sua prática se intensificou na indústria, fazendo com que mudanças nas necessidades dos clientes pudessem ser respondidas com maior facilidade (MAREK; ELKINS; SMITH, 2001).

A utilização do *Kanban* por equipes de desenvolvimento de software começou a ser difundida em 2007, quando resultados obtidos com a aplicação desse método foram apresentados em duas conferências do ramo (ANDERSON, 2010). Neste contexto, executar essa abordagem significa, resumidamente, que somente funcionalidades requisitadas pelo cliente devem ser desenvolvidas pelo time. A adoção do *Kanban* como um ponto de partida para melhoria de processos de desenvolvimento de software vem crescendo desde então (RAJU, KRISHNEGOWDA, 2013) (ANDERSON, 2010).

O *Kanban* continua sendo amplamente utilizado em diferentes setores, também com o objetivo de identificar impedimentos em fluxos de trabalho e reconhecer oportunidades de melhoria contínua (GROSS et al., 2003).

4.2 Definições

De acordo com Gross (2003), o *Kanban* é um sinal visual que determina ao operário o quanto ele deve acelerar e quando ele deve parar o processo produtivo, evitando fabricação superior à demanda existente. Além disso, suas regras indicam como ele deve proceder quando são identificados problemas e quem tem a responsabilidade de solucioná-los.

Anderson (2010), que foi o pioneiro no uso do *Kanban* no desenvolvimento de software, afirma que ele pode ser utilizado em qualquer situação onde exista a necessidade de limitar a quantidade de itens dentro de um sistema. Na sua opinião, o *Kanban* permite aprimorar o processo, incentivando a análise dos problemas e descoberta de soluções.

Segundo Raju e Krishnegowda (2013), o *Kanban* é um método para fazer melhorias incrementais e evolutivas no processo de desenvolvimento de software, gerando redução de custos, melhoria da qualidade, aumento da produtividade e menor tempo de espera.

Enfatizando que o *Kanban* é um conceito relativamente novo em TI, Boeg (2011) o define como um método de gestão de mudanças que permite a visualização e otimização de fluxos de trabalho. Ele destaca que o *Kanban* é construído sobre o conceito de melhoria contínua, sendo ideal quando organizações não pretendem realizar modificações radicais.

4.3 Funcionamento

Resumidamente, o funcionamento desta técnica pode ser descrito da seguinte maneira: um número de cartões representa a capacidade de serviço que o sistema pode colocar em circulação. Cada cartão é atribuído a um item de trabalho que funciona como um mecanismo de sinalização. Um novo item de trabalho só pode ser iniciado no sistema quando um cartão está disponível. Se não há cartões disponíveis, nenhum trabalho adicional pode ser iniciado. Se há cartões disponíveis, eles são atrelados a um item de trabalho e seguem o fluxo do sistema. Qualquer novo trabalho precisa esperar em fila até um cartão estar disponível. Quando um trabalho é finalizado, o cartão é liberado e reciclado. Com um novo cartão disponível, um novo item de trabalho que estava na fila pode ser iniciado (ANDERSON, 2010).

A operação apresentada anteriormente mostra o porquê de o *Kanban* ser caracterizado como um *pull process*, que pode ser traduzido como Processo Puxado. Uma atividade só pode ser efetuada quando há disponibilidade para cumpri-la, então ela é puxada para o processo, em vez de ser "empurrada" com o intuito de seguir um cronograma (TURNER et al., 2012).

4.4 Princípios

Para obter sucesso na implementação do *Kanban*, é essencial compreender seus cinco princípios fundamentais (ANDERSON, 2010):

- Visualizar o fluxo de trabalho;
- Limitar o trabalho em andamento;
- Medir e controlar o fluxo;
- Deixar as políticas do processo explícitas;

- Usar Modelos para reconhecer oportunidades de melhoria.

É importante ressaltar que, antes de colocá-los em prática, deve-se entender como é o funcionamento do sistema no qual se deseja aplicar o *Kanban* (BOEG, 2011). Todo o fluxo de trabalho deve ser mapeado, mas sem o intuito de que ele se torne estático na organização, afinal, uma das características do *Kanban* é a resistência mínima a mudanças (ANDERSON, 2010).

O primeiro princípio mencionado, "Visualizar o fluxo de trabalho", permite que sejam observadas as alterações de status de uma atividade, que pode, por exemplo, mudar da situação "não iniciada" para "finalizada". A necessidade de visualizar este fluxo aumenta ainda mais quando se trata de um processo de alta complexidade, como, por exemplo, um *Kanban* com diversas etapas definidas (KLIPP, 2013).

Equipes frequentemente optam por utilizar um Quadro *Kanban* para visualizar o fluxo de trabalho, pois ele permite um rápido entendimento do estado atual das atividades (TURNER et al., 2012). Ao implantá-lo, cada membro da equipe pode identificar todos os itens de trabalho do processo e quem é o responsável por cada um deles. Outra vantagem que a visualização do quadro possibilita é a auto-organização dos integrantes do time, que podem escolher um item disponível e iniciar sua execução (RAJU; KRISHNEGOWDA, 2013). A Figura 6 representa um exemplo de um quadro *Kanban*.

Ao observar a Figura 6, pode ser identificado o segundo princípio: "Limitar o trabalho em andamento". Os números encontrados logo abaixo ao nome de cada estágio do quadro representam o número máximo de *work in progress* (WIP) ou itens de trabalho em progresso que cada um deles pode abrigar. Este limite é estabelecido para o controle do fluxo, podendo gerar uma redução da sobrecarga que afeta equipes responsáveis por várias tarefas simultâneas (TURNER et al., 2012).

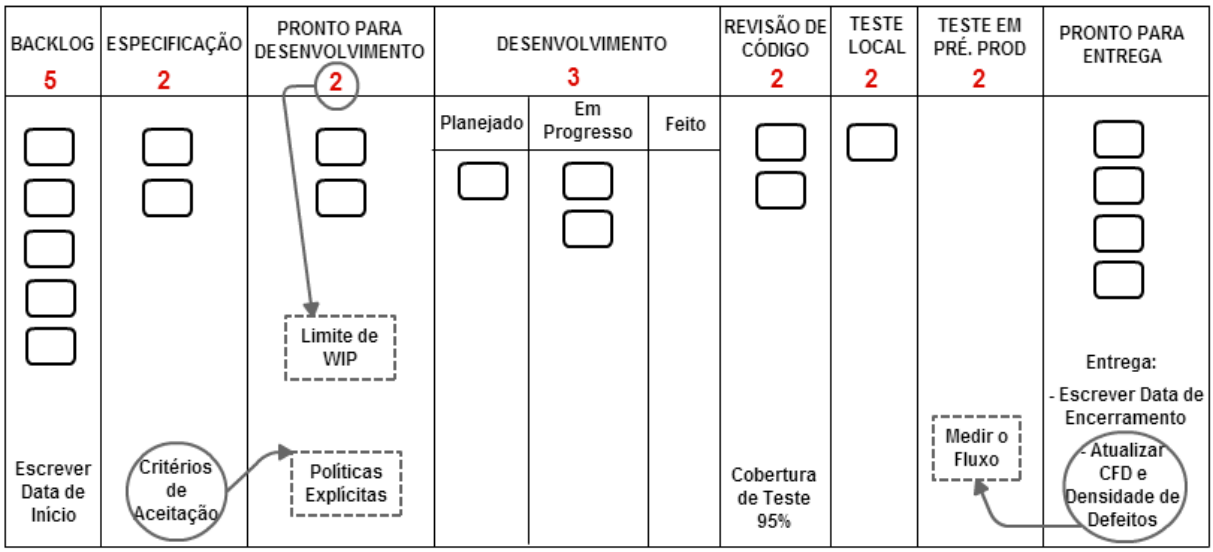

Figura 6 - Quadro Kanban. Fonte: (BOEG, 2011, traduzido)

Anderson (2010) acredita que não há uma fórmula mágica para se chegar ao limite ideal de WIP. O número precisa ser selecionado e a sua utilização regularmente observada. Se ele não estiver adequado, basta ajustá-lo empiricamente. O que não se deve fazer é gastar muito tempo inicialmente tentando determinar o número perfeito. Outro ponto importante é que ele deve ser construído a partir de um consenso da equipe, evitando que, com o passar do tempo, seja substituído ou deixe de ser adotado.

De acordo com Klipp (2013), existe um limite de atividades que podem ser realizadas e ainda executadas satisfatoriamente, sendo ele, muitas vezes, menor do que o imaginado. Ele afirma que, independentemente da complexidade do projeto e do tamanho do time, há uma quantidade ideal de WIP que não compromete a eficiência do processo.

Boeg (2011) aborda que o limite inicial deve ser definido com certa folga, possibilitando que o fluxo presente siga sem impedimentos. Logo após o reconhecimento dos gargalos do processo, este número pode ser ajustado de forma gradual. Finalmente, quando os limites de WIP forem estabelecidos, o sistema só poderá funcionar conforme a capacidade. Um assunto crucial é que não se deve focar somente na determinação da quantidade máxima de itens em progresso, sendo também de suma importância o tamanho que cada um deles apresenta. Tarefas grandes bloqueiam recursos gerando interrupções no fluxo, enquanto tarefas menores fluem pelo sistema fornecendo rápidos feedbacks.

O terceiro princípio, “Medir e controlar o fluxo”, é aplicado para verificar se o caminho certo está sendo seguido. Se uma métrica é utilizada sem um propósito, é provável que ela não devesse estar sendo empregada (BOEG, 2011). Por isso, o *Kanban* precisa promover melhorias que se baseiam em medidas objetivas, de modo a gerar informações significantes que permitam otimização do fluxo e maximização da eficiência (KLIPP, 2013). Algumas dessas medidas são:

- **Diagrama de Fluxo Cumulativo**

O Diagrama de Fluxo Cumulativo (CFD) é considerado por Anderson (2010) uma métrica fundamental, pois indica se o sistema *Kanban* está operando de forma correta. Ele mostra a quantidade de trabalho em cada etapa no processo. A Figura 7 apresenta um exemplo de um CFD:

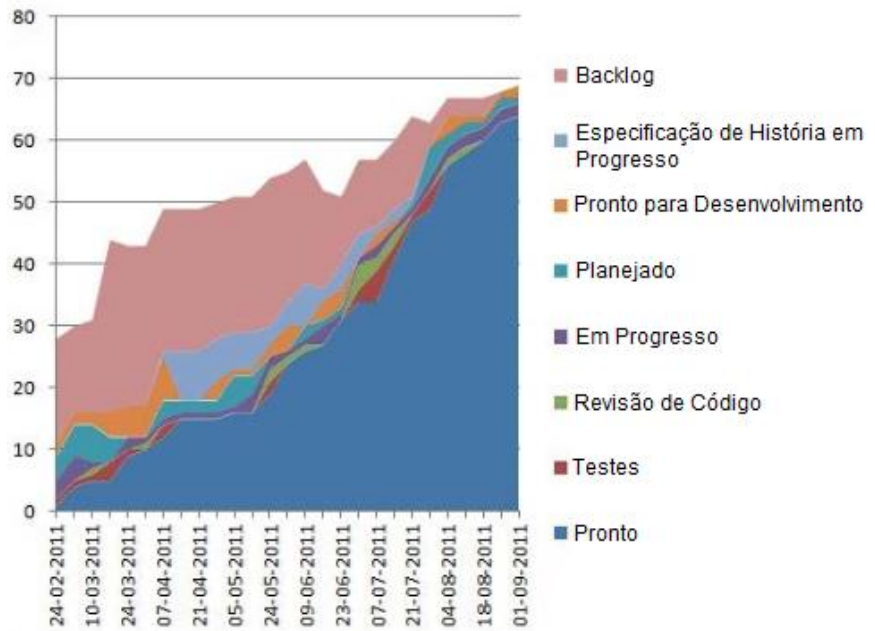

Figura 7 - Exemplo de Gráfico de Fluxo Cumulativo. Fonte: (BOEG, 2011, traduzido)

Uma das várias informações que podem ser obtidas a partir desse gráfico é o trabalho em progresso (WIP), que corresponde à área entre as linhas "Pronto" e "Backlog". Se há uma distância considerável entre essas duas curvas, pode haver um gargalo. Uma linha de "Backlog" muito inclinada em comparação à de "Pronto", por exemplo, indica uma adição de trabalho superior à capacidade de entrega. Outra forma de análise é estimar a data de finalização de um projeto, prevendo o ponto em que essas duas linhas irão se encontrar (BOEG, 2011).

- **Cycle Time e Lead Time**

O *Cycle Time* representa o tempo medido desde que um trabalho é iniciado até a sua conclusão, enquanto o *Lead Time* é calculado a partir do momento em que o pedido é efetuado até a sua entrega. O primeiro possui maior importância na perspectiva do desenvolvedor, pois exibe a capacidade do processo. O segundo é significante para o cliente, que pode identificar o tempo que sua solicitação levou para ser atendida (RAJU; KRISHNEGOWDA, 2013).

- **Índices de Defeitos**

Um meio de verificar a qualidade é calcular o índice de defeitos nos sistemas. Ao detectar e analisar o aumento ou diminuição da incidência dessas falhas no projeto, diversas oportunidades de melhorias podem ser identificadas (BOEG, 2011).

O quarto princípio fundamental para o sucesso da implementação do *Kanban* é "Deixar as políticas do processo explícitas". Muitas equipes de desenvolvimento de software não estabelecem como os distintos tipos de tarefas devem ser tratados. A menos que o processo esteja completamente difundido na organização, é preciso defini-lo e divulgá-lo para todos os envolvidos, a fim de que eles dominem a maneira como o trabalho é efetuado (RAJU; KRISHNEGOWDA, 2013).

Segundo Anderson (2010), é importante pensar em um processo como um conjunto de políticas que governam o comportamento. A existência de regras explícitas proporciona maior facilidade no gerenciamento do fluxo de trabalho, além de possibilitar discussões entre os integrantes, podendo gerar a implantação de diversas melhorias (RAJU; KRISHNEGOWDA, 2013).

Políticas podem ser determinadas para todas as etapas presentes no *Kanban*. Alguns exemplos de políticas estabelecidas estão relacionados ao cumprimento do limite de trabalho em progresso (WIP), à medição do *Cycle Time* e ao cálculo do Índice de Defeitos. Muitas delas são criadas para garantir qualidade e, se elas forem infringidas, o processo pode rapidamente entrar em degeneração (BOEG, 2011).

Por fim, o quinto princípio, "Usar Modelos para reconhecer oportunidades de melhoria", estabelece que métodos devem ser estudados e aplicados visando mudanças contínuas, incrementais e evolutivas no processo (ANDERSON, 2010). Diversos modelos vêm sendo combinados com a metodologia *Kanban* para tratar suas lacunas e promover inovações (TENDON; MULLER, 2013).

Um exemplo de modelo a ser citado é a Teoria das Restrições, filosofia de negócios que se baseia na tomada de decisões considerando as limitações organizacionais. Combiná-la com o *Kanban* pode trazer mais previsibilidade do comportamento do sistema como um todo. Agregando as vantagens das duas técnicas, há uma maior chance de encontrar gargalos no fluxo de trabalho e de aumentar a produtividade da equipe (TENDON; MULLER, 2013).

Os cinco pilares descritos foram constatados a partir da experiência prática em projetos. Ao implementá-los fica muito fácil criar uma cultura "Kaizen" na organização, que nada mais é que frisar a melhoria contínua como responsabilidade de todos (BOEG, 2011).

4.5 Estudos de Caso

Apesar da abordagem *Kanban* ter sido desenvolvida inicialmente para atender a indústria de manufatura, a sua utilização na área de TI vem se tornando expressiva. Resultados mostram que sua adoção nas áreas de desenvolvimento e manutenção de software tem beneficiado organizações (BOEG, 2011).

Um estudo de caso realizado por Ikonen et al. (2011) visa ampliar o entendimento dos impactos que o *Kanban* gera ao ser aplicado em projetos de software. Para analisar esses efeitos, foi elaborado um *framework* contendo nove aspectos de projeto que, com base na literatura, deveriam ser influenciados pela prática do *Kanban*. O estudo foi realizado em uma Fábrica de Software, onde o time era composto por treze integrantes cuja experiência em programação e projeto variava de dois meses a dois anos.

No início do projeto, a equipe desenvolveu um quadro *Kanban* de acordo com as etapas existentes em seu fluxo de trabalho e definiu o limite de WIP. Cada iteração tinha duração de uma semana, que continha, ao final, a apresentação de versões aos clientes e retrospectivas. Em conjunto com essa abordagem, o time empregava práticas ágeis.

Os métodos de pesquisa escolhidos para avaliar o emprego da abordagem foram: observação direta, gravação de vídeos e entrevistas. Ao aplicá-los a equipe obteve os subsequentes resultados em cada um dos nove pontos pertencentes ao *framework*:

- Documentação: Só foram gerados documentos que agregassem valor ao projeto.
- Solução de Problemas: Problemas não acumulavam já que, logo que surgiam, eles eram solucionados. Como um desenvolvedor só podia realizar uma nova tarefa se ele terminasse a anterior, ao encontrar gargalos, outros integrantes o ajudavam para que o trabalho pudesse fluir novamente.
- Visualização: O quadro *Kanban* permitia que todos os envolvidos soubessem a situação atual do projeto, inclusive sobre problemas existentes no fluxo. Além disso, ele facilitava o entendimento do processo de trabalho por parte do time e motivava à auto-organização dos integrantes.
- Compreender o Todo: A equipe conseguiu atingir este ponto a partir da constante apresentação de versões e discussão de resultados com o cliente, da seleção e execução de uma variedade de tarefas e da exploração e debate sobre as diversas soluções de mercado de projetos similares. No entanto, o estudo não mostrou uma relação direta entre o *Kanban* e o alcance desse aspecto contido no *framework*.

- Comunicação: A interação entre o time era constante, ágil e livre. Todos se sentiam como uma única entidade, o que estimulou o surgimento de uma atmosfera de confiança no âmbito da equipe.
- Abraçando o Método: O *Kanban* se mostrou bastante intuitivo, possibilitando aos envolvidos entender e seguir a metodologia com facilidade.
- *Feedback*: O time realizava feedbacks de forma abundante e frequente. Foram definidos momentos para essa atividade ocorrer no processo de desenvolvimento.
- O Processo de Aprovação: A maneira como as atividades transcorriam entre os envolvidos era simplificada. Foram estabelecidas apenas algumas políticas de aprovação para que as tarefas fluíssem nas etapas do quadro, sendo que, se todos os membros estivessem ocupados, elas deveriam permanecer no estágio corrente.
- Selecionando Itens de Trabalho: Desenvolvedores podiam escolher suas tarefas de forma independente, contanto que a ordem de prioridade determinada fosse seguida. Essa liberdade estimulava a equipe na execução de suas atividades.

De acordo com Ikonen et al. (2011), a influência do *Kanban* no processo de desenvolvimento de software ainda foi pouco investigada. A análise a partir do *framework* criado indicou benefícios consideráveis, incluindo a motivação da equipe e um maior controle sobre as atividades dos projetos. Entretanto, por ser um método de monitoramento relativamente básico, é importante que o *Kanban* seja apoiado por práticas e ferramentas adicionais para a obtenção de resultados ainda melhores.

Outra situação no qual a introdução da metodologia *Kanban* ocasionou diversas vantagens ocorreu dentro do Tribunal de Contas da União (TCU). Entre as competências do órgão se destaca a responsabilidade por julgar as contas de administradores públicos e por fiscalizar a aplicação de recursos da União repassados a estados, ao Distrito Federal e a municípios.

Devido à crescente dependência de tecnologia da informação para manipular e armazenar dados da Administração Pública Federal, o Tribunal criou em 2006 a Secretaria de Soluções de Tecnologia da Informação (STI). O estudo de caso foi realizado dentro dessa Secretaria, mais precisamente no 3° Serviço de Soluções de TI (SESOL3), cuja responsabilidade é manter uma lista especificada de sistemas, incluindo as aplicações relacionadas a processos (E-proc) e a documentos (E-doc).

Com o objetivo de monitorar e aprimorar o processo de trabalho, bem como de possibilitar maior visibilidade das atividades executadas pela equipe, o setor optou por aplicar a metodologia *Kanban* como uma forma de gerenciar requisições do cliente. Primeiramente,

foi mapeado todo o processo de trabalho da SESOL3 para se adequar a essa abordagem. Todas as etapas foram discutidas pela equipe e o quadro esboçado foi divido em dois diferentes tipos de requisições: Demanda e Incidente. O primeiro rascunho do painel é apresentado na Figura 8.

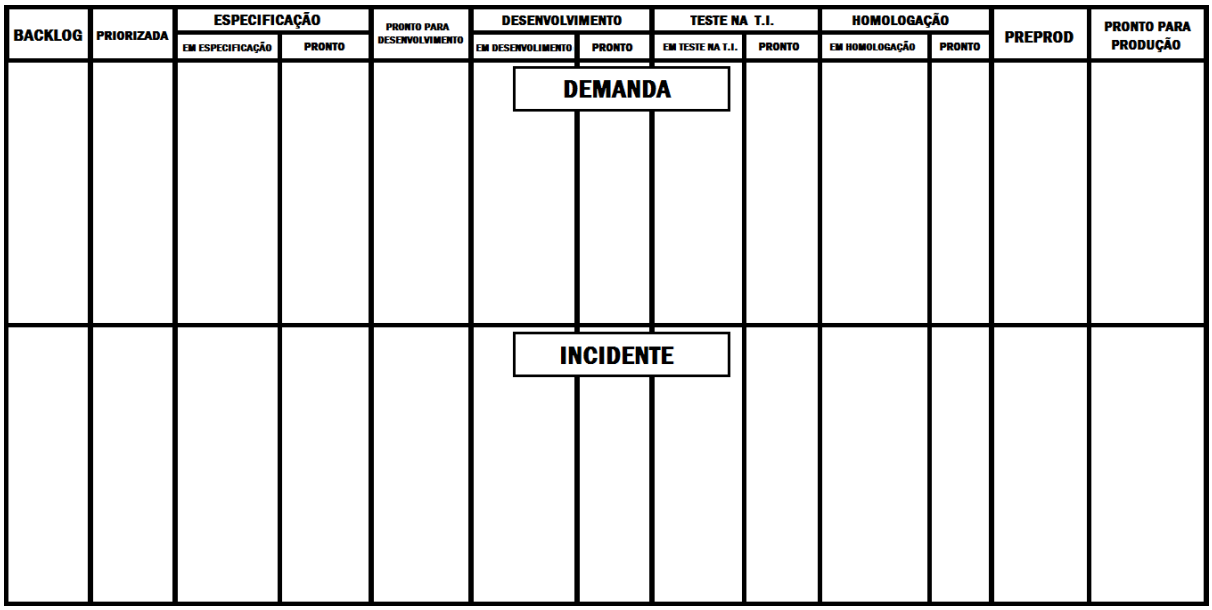

Figura 8 - Esboço do Quadro Kanban. Fonte: (BRASIL, 2015a)

Após o delineamento dos estágios do quadro, foi estabelecido o limite de trabalho em andamento (WIP). Quem definiu este número foi o chefe do setor, considerando a possibilidade de ajustes até que fosse identificado o limite ideal para o processo fluir efetivamente. Políticas também foram impostas, determinando, por exemplo, que o cadastro e entrega de demandas seria quinzenal, enquanto o incidente seria registrado diariamente e disponibilizado a cada cinco dias. A realização de testes funcionais após o desenvolvimento também foi dada como regra no setor.

O primeiro quadro adotado na SESOL-3 foi reproduzido à mão e pode ser visualizado na Figura 9.

Figura 9 - Quadro Kanban Reproduzido à Mão. Fonte: (BRASIL, 2015b)

Durante a utilização do quadro físico, a equipe constatou diversas vezes que era preciso incrementar etapas ou até mesmo alterar seus nomes para que atendessem suas reais necessidades. Esse foi um dos motivos principais para a mudança do primeiro quadro *Kanban*, feito à mão, para um quadro online. A facilidade e rapidez na atualização, visualização de métricas e customização do quadro, além do simples e organizado arquivamento de demandas, concorreram mais ainda para essa substituição.

O *Kanban* se tornou indispensável para a equipe, sendo a visualização do quadro a primeira atividade diária de todos os integrantes do setor. O e-mail que, muitas vezes, era a única forma de comunicação e atualização sobre o que cada integrante estava realizando no momento, não se tornou crucial para que as atividades pudessem ser monitoradas e efetuadas.

5 PROCESSO DE GESTÃO DE DEMANDAS DE MANUTENÇÃO DE SOFTWARE (GEDEM)

Neste capítulo o processo de Gestão de Demandas de Manutenção de Software, denominado GeDeM, é apresentado e detalhado. Inicia-se com a descrição do Ministério, objeto de estudo deste trabalho, assim como uma breve descrição dos fornecedores envolvidos no cenário de desenvolvimento de software do órgão. Em seguida, o processo é detalhado.

5.1 O Ministério

As áreas de competência do Ministério objeto de estudo deste trabalho são os serviços de radiodifusão, postais e de telecomunicações. Em relação ao quantitativo de funcionários da área de TI, o Ministério possui uma força de trabalho de 61 pessoas, sendo 11 servidores (18,03%), 2 administrativos (3,28%) e 48 terceiros (78,69%) (BRASIL, 2014).

Como consequência, o órgão recorre à contratação de serviços de TI e fica responsável pela gestão do contrato. Tradicionalmente, para realizar as contratações, o Ministério segue o Processo de Aquisição de Produtos e Serviços de TI (PAPSTI), apresentado na Figura 10, o qual é alinhado à IN 04/2010 e ao MCTI.

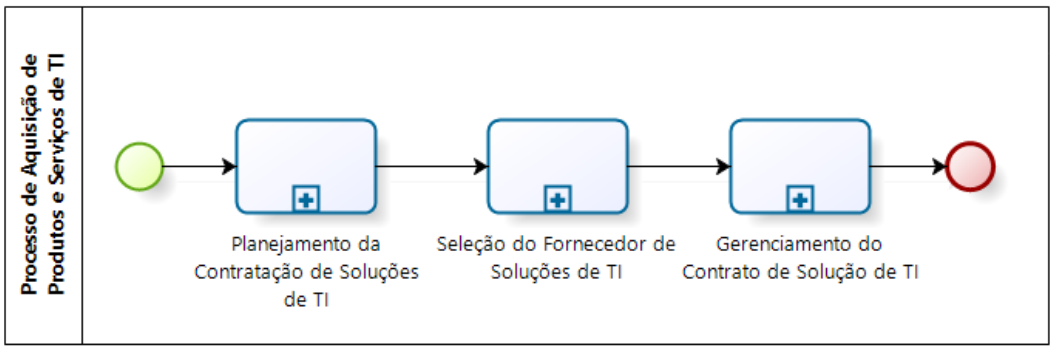

Figura 10 - Processo de Aquisição de Soluções de TI do Ministério. Fonte: (BRASIL, 2012a)

O Ministério também possui a Metodologia de Gestão de Projetos de TI (MGPTI) (BRASIL, 2012b) a fim de padronizar as práticas de gestão de projetos de TI. A MGPTI é aplicada na fase de Gerenciamento do Contrato do PAPSTI e estabelece um ciclo de gerenciamento de projetos flexível, dividido em fases que são definidas de acordo com as características de cada projeto. Após cada fase são realizadas reuniões de decisão que autorizam a passagem do projeto para uma nova fase do seu ciclo de vida (Figura 11).

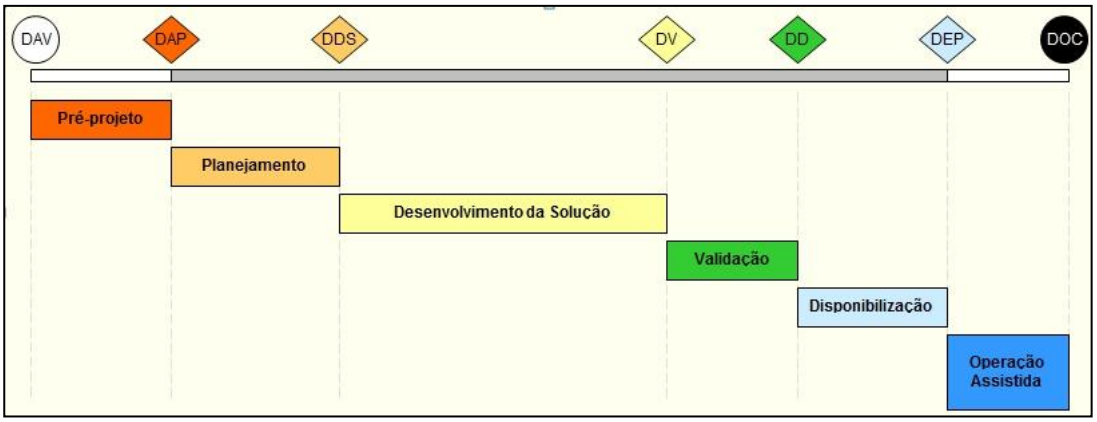

Figura 11 - MGPTI do Ministério. Fonte: (BRASIL, 2012b)

- Decisão de Alinhamento e Viabilidade (DAV): avalia o valor da demanda apresentada para o negócio e autoriza o início de sua análise de viabilidade pela CGTI.
- Decisão de Abertura do Projeto (DAP): autoriza a abertura e o início do planejamento do projeto.
- Decisão de Desenvolvimento da Solução (DDS): avalia o escopo, a solução apresentada e o planejamento para autorizar o início do desenvolvimento da solução.
- Decisão de Validação (DV): avalia se a solução técnica está pronta para o início da validação da solução pelos usuários-chave.
- Decisão de Disponibilização (DD): avalia se a solução técnica tem maturidade para ser implantada e se a organização está preparada para recebê-la.
- Decisão de Encerramento do Projeto (DEP): avalia a disponibilização da solução realizada e autoriza o encerramento do projeto.
- Decisão de Operação Continuada (DOC): avalia a solução em operação em relação aos objetivos de negócio para identificar, se necessário, novas ações de melhoria.

5.2 As Empresas Contratadas

Atualmente, os três contratos mais significativos gerenciados pela área de TI são relacionados à:

- **Fábrica de software:** responsável pela manutenção e desenvolvimento de sistemas;

- **Área de qualidade:** responsável pela validação dos entregáveis pela fábrica de software e verificação da contagem de pontos de função;
- **Infraestrutura de TI**: responsável pela manutenção da infraestrutura de TI do ministério.

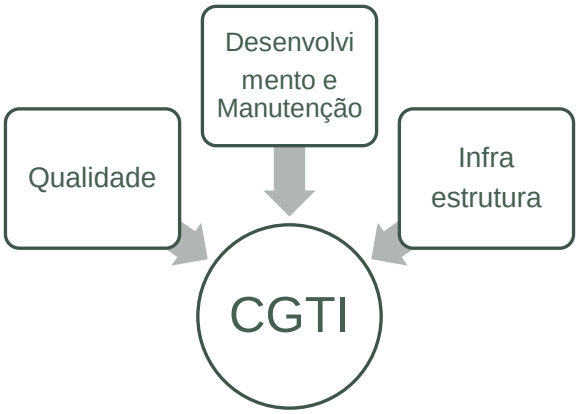

Figura 12 - Contexto das empresas que fornecem serviços de TI para o Ministério. Fonte: autora

A empresa de desenvolvimento e manutenção de sistemas (fábrica de software) é da cidade de Blumenau - Santa Catarina. Dessa empresa, a equipe responsável por desenvolver novos sistemas fica geograficamente distante. O Analista de Requisitos/Líder de Projetos e o Gestor de fábrica viajam esporadicamente para o Ministério. Já a equipe responsável pela manutenção dos sistemas existentes fica alocada no Ministério, juntamente com os funcionários das empresas de qualidade e infraestrutura.

5.3 O Processo GeDeM

O processo de Gestão de Demandas de Manutenção de Software – GeDeM é um dos resultados produzidos no contexto do Termo de Cooperação entre a Universidade de Brasília (UnB) e o Ministérios das Comunicações (MC). Ele é apoiado pela metodologia *Kanban*, tendo em vista que a adoção dessa abordagem nas áreas de manutenção e operação de sistemas provou trazer bons resultados.

5.3.1 Papéis

A seguir são apresentados os papéis do processo GeDeM. São os papéis principais do GeDeM:

- **Usuário**: O Usuário é representado pelo Gestor do Sistema, o qual é detentor das informações relacionadas ao negócio da aplicação. Ele é responsável por solicitar as demandas de manutenção cadastrando-as no quadro *Kanban* e possui papel crucial na homologação de demandas.

- **DISIS**: A Divisão de Desenvolvimento de Sistemas (DISIS) é representada por uma equipe de TI do órgão. Eles devem aprovar as demandas de manutenção a serem desenvolvidas e autorizar a sua implantação em produção.

- **Fábrica de Software**: A Fábrica de Software é responsável por realizar a especificação, a contagem, o desenvolvimento e o teste das demandas de manutenção solicitadas pelo Usuário.

- **Equipe de Qualidade**: A Equipe de Qualidade é responsável por atestar a qualidade do código e de documentos gerados durante a manutenção dos sistemas.

- **Analista de Métricas**: O Analista de Métricas é responsável por realizar a contagem de pontos de função sobre as demandas, além disso ele é responsável por manter atualizado a *baseline* da Contagem APF (Pontos de Função) e o Catálogo BDGC. Suas atividades não são apresentadas na modelagem, pois elas são executadas após a Implantação da Demanda.

- **Infra**: A Infra é representada pela empresa de infraestrutura contratada pelo Ministério, cuja responsabilidade neste processo é implantar o que foi modificado no ambiente de produção.

5.3.2 Processo

A seguir é apresentado o processo GeDeM utilizando a linguagem BPM (*Business Process Management*) para ilustração.

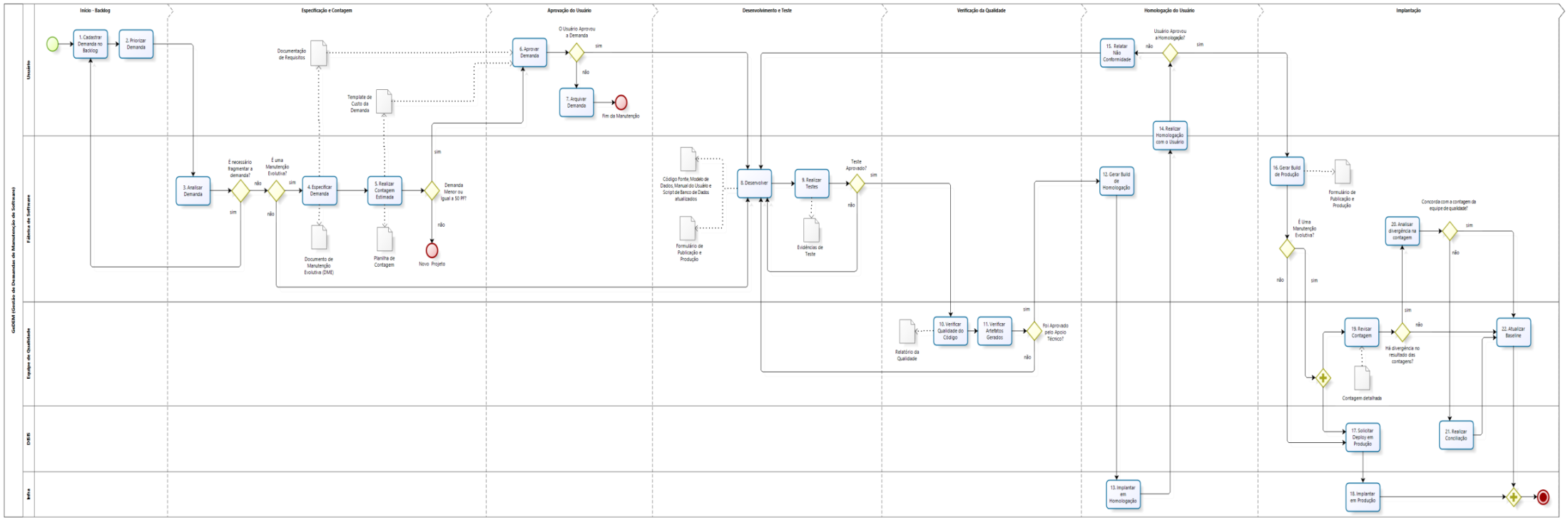

Figura 13 – Processo GeDeM

5.3.3 Matriz de Artefatos

A seguir é apresentada a matriz de artefatos, a qual mapeia a relação entre os artefatos resultantes do processo às etapas do quadro *Kanban* do processo GeDeM.

	Backlog	Especificação e Contagem	Aprovação da Demanda	Desenvolvimento	Testes	Verificação da Qualidade	Homologação do Usuário	Implantação
Documento de Manutenção Evolutiva (DME)		Criado						
Documentação de Requisitos		Criado ou Atualizado		Atualizado				
Planilha de Contagem		Criado						
Template de Custos		Criado						
Código Fonte e Modelo de Dados				Atualizado				
Manual do Usuário				Criado ou Atualizado				
Script de Banco de Dados				Criado ou Atualizado				
Evidências de Teste					Criado			
Relatório de Qualidade						Criado		
Formulário de Publicação e Produção				Criado	Atualizado		Atualizado	

5.3.4 Descrição das Atividades

Nas tabelas a seguir, são apresentadas as atividades do processo GeDeM. São detalhados os responsáveis pela atividade, os artefatos de entrada e saída, uma breve descrição da atividade e os resultados da atividade no quadro *Kanban*.

Tabela 1 - Atividade 1: Cadastrar Demandas no Backlog

Atividade 1: Cadastrar Demandas no Backlog	
Responsável:	Usuário
Artefato de entrada:	N/A
Descrição:	O Usuário cadastra as demandas de manutenção no *Backlog* do quadro *Kanban*.
Resultado no *Kanban:*	Lista das demandas cadastradas no quadro *Kanban,* na coluna ***Backlog*** e subcoluna **Em Andamento**.
Artefato de saída:	N/A

Tabela 2 - Atividade 2: Priorizar Demanda

Atividade 2: Priorizar Demanda	
Responsável:	Usuário
Artefato de entrada:	N/A
Descrição:	O Usuário prioriza as demandas cadastradas no *Backlog* conforme as suas necessidades.

Resultado no *Kanban:*	Lista de demandas priorizadas no quadro *Kanban*, na coluna ***Backlog*- Pronto**. As demandas de maior prioridade devem ser posicionadas acima das de menor prioridade. Neste momento o tempo de resolução da demanda começa a ser contabilizado.
Artefato de saída:	N/A

Tabela 3 - Atividade 3: Analisar Demanda

Atividade 3: Analisar Demanda	
Responsável:	Fábrica de Software
Artefato de entrada:	N/A
Descrição:	A Fábrica de Software deve verificar os seguintes pontos: ○ Se a descrição da demanda é suficiente para a elaboração de sua especificação e sua implementação. Se forem necessárias mais informações sobre a demanda a equipe deverá entrar em contato com o Usuário para que ele aprimore sua descrição; ○ Se a granularidade da demanda está de acordo ou se devem ser criadas novas demandas a partir dela; *Obs*: Se a demanda apresentar uma granularidade maior do que o esperado, a Fábrica de Software deve orientar o Usuário para alterar o nome e descrição da demanda original e criar as demais demandas necessárias.
Resultado no *Kanban*:	A Fábrica de Software puxa a demanda da coluna ***Backlog* – Pronto** para a coluna **Especificação e Contagem – Em Andamento**. Caso a demanda não esteja adequada, de acordo com descrição da atividade, ela deve ser reposicionada na coluna ***Backlog* – Em Andamento** e adicionado um comentário contendo as informações necessárias.
Artefato de saída:	N/A

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

ITRAC

Tabela 4 - Atividade 4: Especificar Demanda

Atividade 4: Especificar Demanda	
Responsável:	Fábrica de Software
Artefato de entrada:	N/A
Descrição:	No caso das demandas de manutenção evolutiva, a Fábrica de Software deve elaborar ou atualizar a especificação relacionada a ela como, por exemplo, o documento de casos de uso.
Resultado no *Kanban*:	A Fábrica de Software, ao finalizar a especificação, não movimenta a demanda localizada em **Especificação e Contagem – Em Andamento**, já que ainda não realizou a atividade de contagem estimada.
Artefato de saída:	○ Documento de Manutenção Evolutiva (DME). ○ Documentação de Requisitos.

Tabela 5 - Atividade 5: Realizar Contagem Estimada

Atividade 5: Realizar Contagem Estimada	
Responsável:	Fábrica de Software
Artefato de entrada:	○ Documento de Manutenção Evolutiva (DME). ○ Documentação de Requisitos atualizada.
Descrição:	Após a especificação da demanda ser finalizada, a Fábrica de Software deve realizar a contagem estimada dos pontos de função decorrentes da manutenção e preencher o *Template* de Custos da Demanda. *Obs:* Além da estimativa do custo total da manutenção, o custo associado à realização da contagem e da especificação, já realizadas pela Fábrica de Software, também deve ser contabilizado e apresentado no *Template* de Custos. Caso a demanda apresente mais do que 50PF, cabe a DISIS decidir se ela deve ser fragmentada em mais demandas e seguir o fluxo de manutenções evolutivas ou se deve ser criado um Novo Projeto.
Resultado no	A Fábrica de Software realiza a contagem e, ao finalizá-la, puxa a

***Kanban*:**	demanda para **Especificação e Contagem – Pronto.**
Artefato de saída:	○ Planilha de Contagem ○ Template de Custos da Demanda

Tabela 6 - Atividade 6: Aprovar Demanda

Atividade 6: Aprovar Demanda	
Responsável:	Usuário
Artefato de entrada:	○ Documentação de Requisitos ○ Template de Custos da Demanda
Descrição:	O Usuário deverá verificar a especificação e os custos associados à demanda. Cabe a ele aprovar ou não, a partir da análise de seu orçamento.
Resultado no *Kanban*:	O Usuário puxa a demanda de **Especificação e Contagem – Pronto** para **Aprovação da Demanda – Em Andamento.** Caso o Usuário aprove a manutenção, ele deve puxar a demanda para **Aprovação da Demanda – Pronto**.
Artefato de saída:	N/A

Tabela 7 - Atividade 7: Arquivar Demanda

Atividade 7: Arquivar Demanda	
Responsável:	Usuário
Artefato de entrada:	N/A
Descrição:	Caso o Usuário decida que a manutenção não deverá ser desenvolvida, tendo em vista o seu orçamento, a demanda será arquivada por ele.
Resultado no *Kanban*:	O Usuário deve arquivar a demanda que se localiza em **Aprovação da Demanda – Em Andamento.**
Artefato de saída:	N/A

Tabela 8 - Atividade 8: Desenvolver

Atividade 8: Desenvolver	
Responsável:	Fábrica de Software
Artefato de entrada:	○ Documentação de Requisitos
Descrição:	Com a demanda aprovada, a Fábrica de Software inicia o desenvolvimento do que foi especificado.
Resultado no *Kanban*:	A Fábrica de Software puxa a demanda de **Aprovação da Demanda – Pronto** para **Desenvolvimento – Em Andamento.** Após desenvolver e realizar os testes unitários e funcionais da demanda, a Fábrica de Software deve puxá-la para **Desenvolvimento – Pronto**.
Artefato de saída:	○ Código Fonte ○ Documento de Requisitos atualizado ○ Modelo de Dados atualizado ○ Script de bando de dados atualizado ○ Formulário de Publicação e Produção

Tabela 9 - Atividade 9: Realizar Testes

Atividade 9: Realizar Testes	
Responsável:	Fábrica de Software
Artefato de entrada:	○ Código Fonte
Descrição:	Após o desenvolvimento da demanda, é necessário realizar o teste de integração para verificar se a implementação está correta.
Resultado no *Kanban*:	A Fábrica de Software puxa a demanda de **Desenvolvimento – Pronto** para **Testes – Em Andamento.** Após realizar os testes com sucesso, a Fábrica de Software deve puxá-la para **Testes – Pronto**. Caso contrário, a demanda deve ser reposicionada na coluna **Desenvolvimento – Em Andamento** e adicionado um comentário justificando a decisão.
Artefato de saída:	○ Evidências de Teste ○ Formulário de Publicação e Produção (Atualizado)

Tabela 10 - Atividade 10: Verificar Qualidade do Código

Atividade 10: Verificar Qualidade do Código	
Responsável:	Equipe de Qualidade
Artefato de entrada:	○ Código Fonte ○ Documentação de Requisitos ○ Evidências de Teste
Descrição:	O Apoio Técnico verifica se o código foi implementado seguindo boas práticas de programação, garantindo que o código entregue atenda aos parâmetros de qualidade.
Resultado no *Kanban*:	O Apoio Técnico puxa a demanda **Testes – Pronto** para **Verificar Qualidade – Em Andamento** e realiza a verificação de código.
Artefato de saída:	○ Relatório de Qualidade

Tabela 11 - Atividade 11: Verificar Qualidade dos Artefatos Gerados

Atividade 11: Verificar Qualidade dos Artefatos Gerados	
Responsável:	Equipe Qualidade
Artefato de entrada:	○ Todos os Documentos Alterados durante o desenvolvimento
Descrição:	O Apoio Técnico verifica se os artefatos relativos à demanda foram devidamente elaborados.
Resultado no *Kanban*:	O Apoio Técnico puxa a demanda **Verificar Qualidade – Em Andamento** para **Verificar Qualidade – Pronto** ao terminar e aprovar as duas verificações (documentos e artefatos). Caso o código ou os artefatos não sejam aprovados, a demanda deve ser reposicionada em **Desenvolvimento – Em Andamento** e adicionar um comentário justificando a decisão**.**
Artefato de saída:	○ Relatório de Qualidade (Atualizado)

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

Tabela 12 - Atividade 12: Gerar build de homologação

Atividade 12: Gerar *build* de homologação	
Responsável:	Fábrica de Software
Artefato de entrada:	○ Código Fonte
Descrição:	Após a demanda ter sido devidamente implementada e testada e com o aval do Apoio Técnico, a Fábrica de Software deverá gerar a *build* de homologação.
Resultado no *Kanban*:	A Fábrica de Software puxa a demanda de **Verificar Qualidade – Pronto** para **Homologação – Em Andamento** e solicita a implantação em ambiente de Homologação do código gerado.
Artefato de saída:	○ Build de Homologação

Tabela 13 - Atividade 13: Implantar em Homologação

Atividade 13: Implantar em Homologação	
Responsável:	Infra
Artefato de entrada:	○ *Build* de Homologação
Descrição:	Com a *build* publicada, a Infra deverá implantá-la em ambiente de homologação.
Resultado no *Kanban*:	A demanda permanece em **Homologação – Em Andamento.**
Artefato de saída:	N/A

Tabela 14 - Atividade 14: Realizar Homologação com o Usuário

Atividade 14: Realizar Homologação com o Usuário	
Responsável:	Fábrica de Software e Usuário
Artefato de entrada:	○ *Build* de Homologação
Descrição:	Com a *build* de homologação publicada, a Fábrica de Software deverá entrar em contato com o Usuário para que seja realizada a homologação assistida. Essa atividade poderá ser feita presencialmente, em uma reunião entre a Fábrica de Software e o Usuário.
Resultado no *Kanban*:	A demanda permanece em **Homologação – Em Andamento**.
Artefato de saída:	N/A

Tabela 15 - Atividade 15: Relatar Não Conformidade

Atividade 15: Relatar Não Conformidade	
Responsável:	Usuário
Artefato de entrada:	N/A
Descrição:	Se durante a homologação o Usuário encontrar alguma não conformidade em relação a demanda solicitada, ele deve relatar para a Fábrica de Software que deve implementar as correções necessárias. É importante ressaltar que as não conformidades relatadas estejam no escopo da demanda, não devendo implicar na criação de novas.
Resultado no *Kanban*:	Caso o Usuário não aprove a manutenção executada, a Fábrica de Software deve reposicionar a demanda em **Desenvolvimento – Em Andamento** e adicionar um comentário justificando a decisão.
Artefato de saída:	N/A

Tabela 16 - Atividade 16: Gerar Build de Produção

Atividade 16: Gerar Build de Produção	
Responsável:	Fábrica de Software
Artefato de entrada:	○ Código Fonte
Descrição:	Após a demanda ter sido homologada e aprovada pelo Usuário, a Fábrica de Software gera a *build* de produção.
Resultado no *Kanban*:	A Fábrica de Software puxa a demanda de **Homologação – Em Andamento** para **Homologação – Pronto**, após gerar a *build* de produção.
Artefato de saída:	○ Formulário de Publicação e Produção (Atualizado)

Tabela 17 - Atividade 17: Solicitar Deploy em Produção

Atividade 17: Solicitar *Deploy* em Produção	
Responsável:	Fábrica de Software
Artefato de entrada:	Formulário de Publicação e Produção
Descrição:	A Fábrica de Software deverá solicitar o *Deploy* em produção. Para isso, ela deve atualizar o formulário de publicação e produção se necessário, colher as assinaturas requeridas e encaminhar à DISIS.
Resultado no *Kanban*:	A Fábrica de Software puxa a demanda de **Homologação – Pronto** para **Implantação – Andamento**.
Artefato de saída:	Formulário de Publicação e Produção atualizado

Tabela 18 - Atividade 18: Implantar em Produção

Atividade 18: Implantar em Produção	
Responsável:	Infra
Artefato de entrada:	N/A
Descrição:	Com a solicitação feita pela Gestão de Mudanças, a Infra implantará a manutenção efetuada no ambiente de produção. Paralelo à implantação, o **Analista de Métricas** deve atualizar a *baseline* da Contagem de APF (Pontos de Função) e o Catálogo BDGC (Campo de Tamanho Funcional na ferramenta OTRS).
Resultado no *Kanban*:	A DISIS puxa a demanda de **Implantação – Em Andamento** para **Implantação – Pronto**, após a Infra ter implantado em ambiente de produção.
Artefato de saída:	N/A

Tabela 19 - Atividade 19: Revisar Contagem

Atividade 19: Revisar Contagem	
Responsável:	Equipe de Qualidade
Artefato de entrada:	○ Planilha de Contagem Detalhada
Descrição:	O Analista de Métricas realiza a recontagem da demanda com o objetivo de aferir o tamanho funcional da manutenção. Adicionalmente, ele deve verificar se houve alguma divergência na contagem de pontos de função da **Fábrica de Software** com a contagem realizado por ele.
Resultado no *Kanban*:	A demanda permanece em **Implantação – Em Andamento**.
Artefato de saída:	○ Planilha de Contagem atualizada

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

Tabela 20 - Atividade 20: Analisar Divergência na Contagem

Atividade 20: Analisar Divergência na Contagem	
Responsável:	Fábrica de Software
Artefato de entrada:	○ Planilha de Contagem atualizada
Descrição:	A Fábrica de Software deve analisar a contagem de pontos de função realizada pela Equipe de Qualidade e avaliar (de forma positiva ou negativa) a divergência das contagens feitas. Caso a Fábrica aceite, mantem-se a contagem efetuada pela Equipe de Qualidade, caso contrário, uma reunião de conciliação deve ser executada.
Resultado no *Kanban*:	A demanda permanece em **Implantação – Em Andamento**.
Artefato de saída:	N/A

Tabela 21 - Atividade 21: Realizar Conciliação

Atividade 21: Realizar Conciliação	
Responsável:	DISIS
Artefato de entrada:	○ Planilha de Contagem Detalhada (Fábrica de Software) ○ Planilha de Contagem atualizada (Equipe de Qualidade)
Descrição:	A DISIS deve convocar os responsáveis pelas contagens e realizar uma reunião para acordar a contagem de pontos de função da manutenção. Deve-se, ao final da reunião, identificar qual das contagens será utilizada para atualização da *baseline.*
Resultado no *Kanban*:	A demanda permanece em **Implantação – Em Andamento**.
Artefato de saída:	N/A

Tabela 22 - Atividade 22: Atualizar Baseline

Atividade 22: Atualizar *Baseline*	
Responsável:	Equipe de Qualidade
Artefato de entrada:	○ Baseline
Descrição:	A Equipe de Qualidade deve atualizar a *Baseline* da contagem no catálogo BDGC.
Resultado no *Kanban*:	A demanda permanece em **Implantação – Em Andamento**.
Artefato de saída:	○ *Baseline* atualizada ○ Catálogo BDGC atualizado

5.3.5 Quadro Kanban

O quadro Kanban apresentado na Figura 14 contém nove etapas e indica os papéis responsáveis por cada uma delas. Apesar de demandas evolutivas e corretivas serem tratadas no processo GeDeM, somente as primeiras serão mantidas no quadro Kanban.

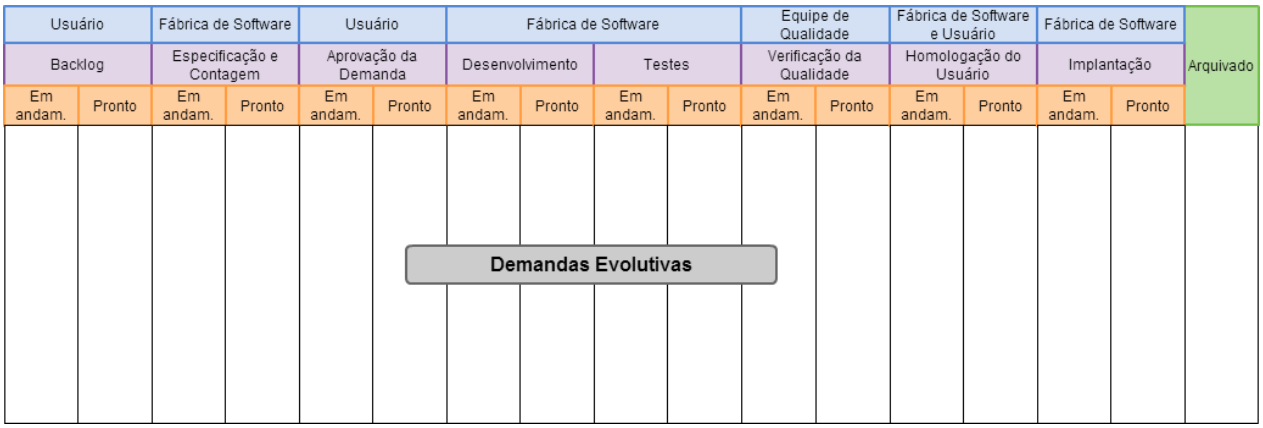

Figura 14 - Quadro Kanban do Processo GeDeM

A primeira linha do quadro, em azul, representa os papéis envolvidos no processo, a linha de cor roxa representa às atividades a serem realizadas e a de cor laranja representa os possíveis estados destas atividades. A última coluna, de cor verde, corresponde ao armazenamento das demandas finalizadas.

Para que uma manutenção seja executada, ela deve ser inicialmente cadastrada em um cartão que irá percorrer todo o quadro. O Kanban é classificado como um processo “puxado”, ou seja, o responsável deve buscar o cartão da etapa anterior e trazê-lo para sua etapa se houver disponibilidade, visando pró-atividade da equipe.

Porém, existem algumas políticas ou regras que devem ser cumpridas no intuito de que isso ocorra. As etapas do Kanban apresentam duas colunas “Em Andamento” e “Pronto”. O cartão só poderá ser movido para a coluna “Pronto” quando todas as políticas da etapa em que ele se encontra forem satisfeitas.

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

5.3.6 Políticas do Quadro *Kanban*

Para alterar o status das demandas no quadro *Kanban* do estado de "*Em Andamento*" para "*Pronto*" em cada coluna, as seguintes políticas devem ser atendidas (elas devem ser respondidas com "*sim*"):

- **Coluna:** Backlog (**Responsável:** Usuário)

1. A demanda está atômica (não agrupa mais de uma demanda)?

2. A demanda está priorizada (as demandas de maior prioridade estão no topo)?

- **Coluna:** Especificação e Contagem (**Responsável:** Fábrica de Software)

1. A análise da demanda foi finalizada (descrição e granularidade adequadas para o desenvolvimento)? Considera-se uma demanda de granularidade adequada aquela que não agrupa mais de uma funcionalidade.

2. Se a demanda foi analisada e foi detectada a necessidade de alterar a descrição/nome da demanda, a Fábrica de Software entrou em contato com o Usuário? A descrição/nome da demanda foi atualizada no cartão?

3. Se a demanda foi analisada e apresentou granularidade inadequada, a Fábrica de Software entrou em contato com o Usuário e o orientou a quebrá-la em demandas menores? Foi feito um comentário do problema identificado e a demanda foi reposicionada na coluna "Backlog"?

4. O Documento de Manutenção Evolutiva (DME) foi finalizado?

5. A Documentação de Requisitos referente à demanda foi criada/atualizada?

6. A Contagem estimada de PF da demanda foi finalizada?

7. A Planilha de Contagem foi atualizada?

8. O Template de Custos foi preenchido?

9. Todos os artefatos criados ou atualizados foram anexados no cartão?

10. Se a demanda for maior que 50 PF, a DISIS foi acionada e foi aprovada a fragmentação da demanda?

- **Coluna:** Aprovação da Demanda (**Responsável:** Usuário)

1. A demanda foi analisada e seu desenvolvimento foi aprovado pelo Usuário? Se sim, ela foi movida para a coluna “Aprovação do Usuário – Pronto”?

2. Caso a demanda tenha sido rejeitada, ela foi arquivada?

- **Coluna:** Desenvolvimento (**Responsável:** Fábrica de Software)

1. O desenvolvimento da demanda foi finalizado?

2. Os testes unitários desenvolvidos passaram?

3. O modelo de dados foi atualizado?

4. O manual do Usuário foi atualizado?

5. O Formulário de Publicação e Produção foi criado?

6. Todos os artefatos criados ou atualizados foram anexados no cartão?

- **Coluna:** Testes (**Responsável:** Fábrica de Software)

1. Os testes funcionais executados passaram?

2. Os testes de integração realizados sobre a demanda passaram?

3. As evidências de testes foram produzidas?

4. O Formulário de Publicação e Produção foi atualizado?

5. Todos os artefatos criados ou atualizados foram anexados no cartão?

- **Coluna:** Verificação da Qualidade (**Responsável:** Equipe de Qualidade)

1. O código desenvolvido apresentou a qualidade exigida?

2. Os artefatos gerados apresentaram a qualidade exigida?

3. O Relatório de Qualidade foi criado e atualizado?

4. Caso o código e os artefatos gerados não apresentem a qualidade adequada, foi adicionado um comentário do problema identificado e a demanda foi reposicionada na coluna "Desenvolvimento"?

5. Todos os artefatos criados ou atualizados foram anexados no cartão?

- **Coluna:** Homologação (**Responsável:** Fábrica de Software)

1. A build de homologação foi gerada e publicada?

2. A Infra implantou o código em homologação?

3. A homologação assistida (Usuário e Fábrica de Software) foi efetuada?

4. A manutenção executada foi aprovada pelo Usuário?

5. Se a manutenção não foi aprovada pelo Usuário, foi adicionado um comentário identificando o problema encontrado e a demanda foi reposicionada para a coluna "Desenvolvimento"?

6. A build de produção foi gerada e publicada?

7. O Formulário de Publicação e Produção foi finalizado e anexado no cartão?

- **Coluna:** Implantação (**Responsável:** Fábrica de Software)

1. O formulário de publicação e produção contém todas as assinaturas necessárias?

2. O código foi autorizado a ser implantado em produção pela Gestão de Mudanças?

3. A Infra foi informada para a implantação do código em produção?

4. O código foi implantado em produção? Foi adicionado um comentário com a data em que a demanda foi implantada ao cartão do quadro Kanban?

5. A contagem de pontos de função foi acordada entre Fábrica de Software e Equipe de Qualidade?

6. A baseline do catálogo BDGC foi atualizada?

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

ITRAC

6 CONSIDERAÇÕES FINAIS

O presente trabalho tem como finalidade principal solucionar um problema prático vivenciado por um Ministério do Governo Federal. O processo de manutenção de software vigente no órgão não está totalmente definido e os diferentes envolvidos se encontram insatisfeitos com a forma pela qual ele é executado. Além disso, grande parte do conhecimento relativo a esse serviço se concentra nas empresas terceirizadas, dificultando maior controle e monitoramento por parte da área de TI do Ministério em relação ao andamento das atividades pertinentes ao processo.

Para combater esse problema, a Equipe da UnB realizou um diagnóstico no órgão e propôs um novo processo de manutenção de software utilizando a metodologia *Kanban*, o qual foi continuamente aprimorado de modo a obter uma versão adequada para teste em um piloto. Aumentar a visibilidade e o acompanhamento das demandas de manutenção por parte dos envolvidos foi o objetivo central da adoção desse processo.

Foi um desafio para o grupo participante deste trabalho, executar um processo completamente novo, composto de vários papéis, já que se exigiu uma grande mudança cultural no órgão, principalmente devido à utilização da metodologia *Kanban*. Enquanto no processo vigente as demandas de manutenção são empurradas aos responsáveis para seguir um cronograma, na abordagem *Kanban* elas são puxadas, requerendo pró-atividade dos participantes.

A metodologia de pesquisa selecionada para esse estudo, a Pesquisa-Ação, se mostrou ideal para vencer esse desafio e diminuir a resistência dos envolvidos, pois a Equipe da Unb estava sempre próxima, apoiando e colaborando para a execução das atividades. A primeira etapa da Pesquisa-ação, o Diagnóstico, foi realizada somente uma vez para identificar os problemas do processo vigente, enquanto que as três últimas etapas, o Planejamento, a Ação e a Avaliação, ocorreram três vezes, permitindo a aplicação de melhoria nos ciclos seguintes.

O comprometimento de todos os participantes, desde o início do trabalho até a avaliação final, foi essencial para a realização do processo, que transcorreu da melhor forma possível. Vale ressaltar a participação dos Usuários, os quais estavam sempre motivados, contribuindo com muitas sugestões e acreditando no potencial do processo.

Os resultados obtidos nos questionários e com a aplicação das métricas coletadas demonstraram que o novo processo mitigou vários dos problemas constatados anteriormente, mostrando-se mais eficaz e consistente, além de provocar uma maior satisfação dos envolvidos. Quando um gargalo era detectado na conclusão de cada ciclo, participantes realizavam discussões conjuntas para encontrar soluções e identificar oportunidades de melhoria.

Um aspecto que impactou na execução do processo foi a baixa capacidade de atendimento da Equipe de Manutenção, a qual apresentava somente um membro alocado para

solucionar todas as demandas do piloto, já que este trabalho foi realizado em um período de mudança da empresa contratada. Com um número maior de integrantes na Equipe de Manutenção, mais demandas certamente passariam por todas as etapas do processo, gerando mais resultados para avaliar sua eficiência.

Como melhoria futura, a expectativa era que uma nova fábrica ao iniciar seus trabalhos de manutenção no órgão, o WIP seja analisado e estabelecido para todas as colunas do quadro *Kanban*. Outro ponto a ser aprimorado diz respeito a análise e escolha da ferramenta *Kanban* que será utilizada na implantação do processo, após o piloto, para todos os sistemas no Ministério. Apesar da aplicação *Trello* atender requisitos identificados como importantes pela Equipe da UnB, suas informações se encontram na nuvem, o que causa insegurança aos envolvidos. Por conseguinte, o pessoal do MC considerou fundamental que a ferramenta a ser adotada no Ministério seja *standalone*, garantindo que as informações dos sistemas mantidos sejam acessadas exclusivamente no ambiente interno.

Ao final, é válido registrar o *feedback* positivo a respeito deste trabalho por parte dos envolvidos, que elogiaram a iniciativa de aprimorar os serviços de manutenção de software e de contribuir para aumentar a transparência em um órgão público federal. Em uma das reuniões para decisões de melhorias no processo GeDeM, esse posicionamento favorável ficou mais evidenciado com a seguinte fala do atual Chefe da DISIS do MC: “A semente da melhoria já foi plantada, daqui para a frente vamos colher os resultados”.

7 REFERÊNCIAS

ALARANTA, M.; JARVENPAA, S. **Changing IT providers in public sector outsourcing: Managing the loss of experiential knowledge.** 43rd Hawaii International Conference on System Sciences (HICSS), 2010.

ANDERSON, D. **Kanban - Successful Evolutionary Change for your Technology Business**. Blue Hole Press. ISBN 0-9845214-0-2, 2010.

APRIL, A. **Studying Supply and Demand of Software Maintenance and Evolution Services**, 2010.

APRIL, A., ABRAN, A. **Software Maintenance Management**. Wiley, 2008.

BENNETT, K. H., XU, J. **Software Services and Software Maintenance.** *Proceedings of the Seventh European Conference of Software Maintenance and Reengineering*, 2003.

BHATT, P., SHIROFF, G. E MISRA, A. K. **Dynamics of Software Maintenance**, 2004.

BHATTACHARYA, P. **Using Software Evolution History to Facilitate Development and Maintenance**, 2011.

BOEG, J. **Kanban Em 10 Passos**. Leonardo Galvão, 2011.

BRASIL. **Decreto-Lei Nº 200, de 25 de Fevereiro de 1967. Dispõe sobre a organização da Administração Federal, estabelece diretrizes para a Reforma Administrativa e dá outras providências,** 1967. Disponível em: <http://www.planalto.gov.br/ccIVIL_03/Decreto-Lei/Del0200.htm>.

BRASIL. **Lei Nº 8.666, de 21 de junho de 1993**, 1993 Disponível em: <http://www.planalto.gov.br/ccivil_03/leis/l8666cons.htm>.

BRASIL. **Instrução Normativa - SLTI 4, de 19 de maio de 2008. Dispõe sobre o processo de contratação de serviços de Tecnologia da Informação pela Administração Pública Federal direta, autárquica e fundacional,** 2008. Disponível em: <http://www.comprasnet.gov.br/legislacao/in/ in0408.htm>.

BRASIL, Secretaria de Logística e Tecnologia da Informação**. Instrução Normativa No04, de 12 de novembro de 2010. Dispõe sobre o processo de contratação de Soluções de Tecnologia da Informação pelos órgãos integrantes do Sistema de Administração dos Recursos de Informação e Informática (SISP) do Poder Executivo Federal,** 2010. Disponível em: <http://www.governoeletronico.gov.br/biblioteca/arquivos/instrucao-normativa-no-04-de-12-de-novembro-de-2010/download>

BRASIL, Ministério das Comunicações. **Norma Operacional SPOA No006, de 10 de Setembro de 2012. Dispõe sobre a Metodologia de Gerenciamento de Projetos de**

Tecnologia da Informação - MGP -TI, utilizada no âmbito do Ministério das Comunicações, 2012a.

BRASIL. Ministério das Comunicações. **Norma Operacional SPOA No 007, de Setembro de 2012. Dispõe sobre o Processo de Aquisição de Produtos e Serviços de Tecnologia da Informação, utilizado no âmbito do Ministério das Comunicações.,** 2012a.

BRASIL, Tribunal de Contas da União. **Guia de Boas Práticas em Contratação de Soluções de Tecnologia da Informação,** 2012b. Disponível em: <http://www.governoeletronico.gov.br/biblioteca/arquivos/guia-de-boas-praticas-em-contratacao-de-solucoes-de-tecnologia-da-informacao-tcu>.

BRASIL, Secretaria de Logística e Tecnologia da Informação. **Guia Prático para Contratação de Soluções de TI,** 2014a. Disponível em: <http://www.governoeletronico.gov.br/sisp-conteudo/nucleo-de-contratacoes-de-ti/modelo-de-contratacoes-normativos-e-documentos-de-referencia/guia-de-boas-praticas-em-contratacao-de-solucoes-de-ti>

BRASIL. **Estratégia Geral de Tecnologia da Informação e Comunicações 2014-2015 / Ministério do Planejamento, Orçamento e Gestão, Secretaria de Logística e Tecnologia da Informação,** 2014b. Disponível em: <http://www.sti.ufpb.br/documentos/EGTIC.pdf>

BRASIL, Secretaria de Logística e Tecnologia da Informação. **Instrução Normativa Nº 4, de 11 de setembro de 2014. Dispõe sobre o processo de contratação de Soluções de Tecnologia da Informação pelos órgãos integrantes do SISP do Poder Executivo Federal,** 2014c. Disponível em:<http://www.governoeletronico.gov.br/biblioteca/arquivos/instrucao-normativa-nb0-4-de-11-de-setembro-de-2014-compilada/download>

BRASIL, Ministério das Comunicações. **Plano Estratégico de Tecnologia da Informação (PETI) e Plano Diretor de Tecnologia da Informação (PDTI) 2013 - 2015,** 2014d. Disponível em: <http://www.mc.gov.br/legislacao/por-tipo/pdti-mc/plano-diretor-de-tecnologia-da-informacao-do-ministerio-das-comunicacoes-pdti-para-o-periodo-2013-2015>

BRASIL, Tribunal de Contas da União. Sesol-3, 2015a. Disponível em <https://contas.tcu.gov.br/wikiti/images/5/58/Quadro_modelo.png>

BRASIL, Tribunal de Contas da União. Sesol-3, 2015b. Disponível em <https://contas.tcu.gov.br/wikiti/images/c/cf/Qadrokanban_fixo.jpg>

COHN, M. **Succeeding with agile: software development using Scrum.** Upper Saddle River, NJ: Addison-Wesley, 2010.

CRUZ, C. S. d.; ANDRADE, E. L. P. d.; FIGUEIREDO, R. M. DA C. **Processo de Contratação de Serviços de Tecnologia da Informação para Organizações Públicas.**

Área Especial de Industria Projeção A - UnB - Setor Leste Gama - Brasília DF.
CEP: 72444-210 | Caixa Postal: 8114

ITRAC

Brasília: Série de livros: Ministérios da Ciência e Tecnologia. Secr. de Política de Informática, 2011.

GIL, A. C. **Como Elaborar Projetos de Pesquisa.** São Paulo: Atlas, 2008.

GROSS, J. M., MCINNIS, K. R. **Kanban Made Simple: Demystifying and Applying Toyota's Legendary Manufacturing Process.** New York, USA: AMACOM Books, 2003.

GRUBB, P., TAKANG, A. **Software Maintenance: Concepts and Practice.** World Scientific, 2003.

HAVLICE, Z., KUNŠTÁR, J., ADAMUŠČÍNOVÁ, I., PLOČICA, O. **Knowledge in Software Life Cycle**, 2009.

HUNT, B., TURNER, B., MCRITCHIE, K. **Software Maintenance Implications on Cost and Schedule**, 2008.

IKONEN, M., PIRINEN, E., FAGERHOLM F., KETTUNEN, P., ABRAHAMSSON, P. **On the Impact of Kanban on Software Project Work**, 2011.

ISO/IEC. **International Standard ISO/IEC/IEEE 14764 Software Engineering - Software Life Cycle Processes - Maintenance.** Piscataway, EUA, 2006.

KINOSHITA, F. **Practices of an Agile Team**, 2008.

KLIPP, P. **Getting Started With Kanban,** 2013

MAREK, R., ELKINS, D. A., SMITH, D. R. **Understanding the Fundamentals of Kanban and Conwip Pull Systems Using Simulation**, 2001.

MELLO, C. H. P. et al. **Pesquisa-ação na engenharia de produção: proposta de estruturação para sua condução.** Produção, v. 22, n. 1, p. 1–13, 2012.

MORESI, E. **Metodologia da Pesquisa.** Universidade Católica de Brasília, 2003.

OHNO, T. **O Sistema Toyota de Produção de Produção – Além da Produção em Larga Escala.** Bookman, 1997.

OZA, N., FAGERHOLM, F., MUNCH, J. **How Does Kanban Impact Communication and Collaboration in Software Engineering Teams?**, 2013.

PFLEEGER, S. L. **Engenharia de Software: Teoria e Prática.** Prentice Hall, 2004.

PODNAR, I., MIKAC, B. **Software Maintenance Process Analysis Using Discrete-Event Simulation**, 2001.

PRESSMAN, R. S. **Engenharia de software.** Bookman, 2006.

RAJU, H. K., KRISHNEGOWDA, Y.T. **Kanban Pull and Flow – A Transparent Workflow for Improved Quality and Productivity in Software Development**, 2013.

RASHID, A., WANG, W. Y. C., DORNER, D. **Gauging the Differences between Expectation and Systems Support: the Managerial Approach of Adaptive and Perfective Software Maintenance**, 2009.

REN, Y., CHEN, X., XING, T., CHAI, X. **Research on Software Maintenance Cost of Influence Factor Analysis and Estimation Method**, 2011.

SWANSON, E. B. **The Dimension of Maintenance**, 1976.

TENDON, S., MULLER W. TAME THE FLOW. **Hyper-Productive Knowledge-Work Management**, 2013.

THIOLLENT, M. **Metodologia da pesquisa-ação**. Autores Associados, 1986.

THIOLLENT, M. **Pesquisa-ação nas organizações**. São Paulo (SP): Atlas, 2009.

TRIPATHY, P. E NAIK, K. **Software Evolution and Maintenance. A Practitioner's Approach**. Wiley, 2008.

TRIPP, D. **Pesquisa-ação: uma introdução metodológica**, 2005.

TURNER, R., MADACHY, R., LANE, J., IGOLD, D., ANDERSON, D. **An Event driven, Value-based, Pull Systems Engineering Scheduling Approach**, 2012

WOHLIN, C. et al. **Experimentation in Software Engineering**. Berlin Heidelberg New York: Springer-Verlag, 2012.

YONGCHANG, R., TAO, X., ZHONGJING, L. XIAOJI, C. **Software Maintenance Process Model and Contrastive Analysis**, 2011

ANEXO 1 – Produção Acadêmica

Como produção acadêmica, na temática do relatório, apresenta-se uma coletânea dos artigos publicados e uma coletânea dos *Trabalhos de Conclusão de Curso* da Faculdade GAMA – FGA, relacionados e oriundos do Projeto de Pesquisa.

1. Artigos em conferências nacionais e internacionais:

- Noronha, A. P. V.; Venson, E.; Figueiredo, R. M. C.; Modesto, A. S. C. Applying Kanban to Manage Outsourced Maintenance Services: An Action Research in a Brazilian Government Agency. In: CIbSE Conference IberoAmerican on Software Engineering (ESELAW Experimental Software Engineering Track), 22-23 May, 2017, Buenos Aires, Argentina.
  Link: *Indisponível*

- Santos, Jads Victor Paiva dos; Figueiredo, R. M. C; Noronha, Ana Paula Vargas de; Venson, Elaine. Using Kanban in Outsourced Government Projects of Management Maintenance Demands: a Descriptive Research. In: 13th CONTECSI International Conference on Information Systems and Technology Management, 2016. p. 4147
  Link: http://www.contecsi.fea.usp.br/envio/index.php/contecsi/13CONTECSI/paper/view/4204

- Soares, V. A.; Figueiredo, R.; Venson, Elaine; Araujo, L. B.; Rafael Queiroz. "Inventorying Systems: an Action Research", in: International Conference on Enterprise Information Systems (ICEIS), 2017, Porto - Portugal.
  Link: http://www.scitepress.org/DigitalLibrary/PublicationsDetail.aspx?ID=uk10Ff0L2w8=&t=1

- Sousa, T. L. de; Venson, E.; Figueiredo, R. M. C.; Kosloski, R. A.; Ribeiro Júnior, L. C. M. "Using Scrum in Outsourced Government Projects: An Action Research," in 2016 49th Hawaii International Conference on System Sciences (HICSS), 2016, pp. 5447–5456.
  Link: http://ieeexplore.ieee.org/document/7427860/

- Sousa Sobrinho, L. P. de; Figueiredo, R. M. da C.; Venson, E.; Ribeiro Jr, L. C. M.; Souza,T. L. de;Kosloski,R. A. D. "Application of the scrum agile *framework* to the management process of software development outsourcing in a Brazilian Government Agency," in 12o CONTECSI International Conference on Information Systems and Technology Management, 2015.
  Link: http://www.contecsi.fea.usp.br/envio/index.php/contecsi/12CONTECSI/paper/view/3140

- Souza, Thatiany; Figueiredo, R. M. C.; Venson, E.; Kosloski, R. A. D.. Experiência No Projeto *Framework* de Soluções de TI. In: VII Fórum de Educação em Engenharia de Software (FEES 2014), evento integrante do XXVIII Simpósio Brasileiro de Engenharia de Software (SBES 2014), Maceió. AL, 2014.
  Link: http://www.ic.ufal.br/evento/cbsoft2014/anais/fees_v1_p.pdf

- Brito, M F de; Figueiredo, R M C;Venson, E; Canedo, E D.; Ribeiro Jr, L C M. "Knowledge Transfer in a Management Process for Outsourced Agile Software Development", in *50th Hawaii International Conference on System Sciences* (HICSS), 04-07, 2017, Hawaii.
  Link: http://scholarspace.manoa.hawaii.edu/handle/10125/41921

- Morais, Emilie de; Jesus, Geovanni de; Figueiredo, R.; Venson, Elaine; Rafael Queiroz. "Knowledge Transfer in IT Service Provider Transition". In: International Conference on Enterprise Information Systems (ICEIS), 2017, Porto - Portugal.
  Link: http://www.scitepress.org/DigitalLibrary/PublicationsDetail.aspx?ID=sbNJU7kvtOI=&t=1

- Brito, M. F.; Figueiredo, R. M. da C.;Venson, E.; Ribeiro. Jr, L. C. M.; Kosloski, R. A. D.; "Transferência de Conhecimento em Projetos de Desenvolvimento de Software no Contexto de Contratação, " in *12°CONTECSI - InternationalConferenceon Information Systems and Technology Management*, 2015.
  Link: http://www.contecsi.fea.usp.br/envio/index.php/contecsi/12CONTECSI/paper/view/3175

2. Trabalhos de Conclusão de Curso da Faculdade GAMA – FGA, publicados pela Biblioteca FGA:

- **2016/2**

  TCC2 – Implantação de Processos de Inventariação de Software para um Órgão Público Federal Brasileiro: uma pesquisa-ação. Vanessa de Andrade.

- **2016/1**

  TCC1 – Implantação de Processos de Inventariação de Software para um Órgão Público Federal Brasileiro: uma pesquisa-ação. Vanessa de Andrade.

  TCC2 – Ferramenta de Gestão de Contratos de Fábrica de Software para um Órgão Público Brasileiro; Thabata Granja.

- **2015/2**

  TCC1 – Ferramenta de Gestão de Contratos de Fábrica de Software para um Órgão Público Brasileiro. Thabata Granja.

  TCC2 – Uso do Kanban no Tratamento de Demandas de Manutenção de Software: Uma Pesquisa- Ação em um Órgão Público Federal Brasileiro; Ana Paula Vargas de Noronha.

  TCC2 – Processo de Inventariação de Software para um Órgão Público Federal Brasileiro; Laís Barreto de Araújo.

- **2015/1**

  TCC1 - Uso do Kanban no Tratamento de Demandas de Manutenção de Software: Uma Pesquisa-Ação em um Órgão Público Federal Brasileiro; Ana Paula Vargas de Noronha.

  TCC1 - Proposição de um Processo de Catalogação de Softwares Legados em um Órgão Público Federal Brasileiro; Laís Barreto de Araújo.

- **2014/2**

  TCC1 – Monitoração da Qualidade de Produto nas Contratações de Soluções de TI da Administração Pública Federal; Luiza Shaidt e Yago Regis.

  TCC2 - Uso do Scrum na Contratação de Fábrica de Software: Uma Pesquisa-Ação em um Órgão Público Federal Brasileiro; Thatiany Lima.

TCC2 - Uso do Kanban em um Processo de Gestão de Demandas de Manutenção de Software por Terceiros para um Órgão Público Federal; Jads Victor.

- **2014/1**

  TCC1 – Transferência de Conhecimento em Contratação de Fábricas de Software: Uma Pesquisa-Ação em Órgão Público Federal Brasileiro; Thatiany Lima.

  TCC1 - Uso do Kanban em um Processo de Gestão de Demandas de Manutenção de Software por Terceiros para um Órgão Público Federal; Jads Victor.

  TCC2 – Aspectos de Validação de Software em Metodologias Ágeis Aplicáveis a Terceirização do Desenvolvimento de Software; Eduardo Barbosa.

  TCC2 – Uso do Scrum em um Processo de Gestão de Demandas de Desenvolvimento de Software por Terceiros para um Órgão Público Federal Brasileiro; Luiz Pereira de Souza Sobrinho.

  TCC2 - Definição de Critérios de Aceite baseados em Métricas de Software para um Processo Ágil de Gestão de Demandas de Desenvolvimento de Software; Tiago Gomes.

  TCC2 - Uma Proposta de Base Histórica de Medições para Uso em Desenvolvimento Ágil de Software; Breno Dantas.

- **2013/2**

  TCC1 - Validação em Processos de Contratação de Fábrica de Software Baseados nos Princípios Ágeis; Eduardo Barbosa.

  TCC1 – Uso do Scrum em um Processo de Gestão de Demandas de Desenvolvimento de Software por Terceiros para um Órgão Público Federal Brasileiro; Luiz Pereira de Souza Sobrinho.

  TCC1 - Definição de Critérios de Aceite baseados em Mé tricas de Software para um Processo Ágil de Gestão de Demandas de Desenvolvimento de Software; Tiago Gomes.

  TCC2 - Transferência de Conhecimento em Processos de Contratações de Fábricas de Software por Organizações Públicas Federais; Maylon Felix Brito.

- **2013/1**

  TCC1 - Transferência de Conhecimento em Processos de Desenvolvimento de Software Ágeis no Contexto de Contratação; Maylon Felix Brito.