Universidade Federal de Minas Gerais
Escola de Engenharia
Programa de Pós-graduação em
Engenharia de Produção

Carla Floriana Martins

TRADIÇÃO E MODERNIDADE: DES (ENTENDIMENTOS) NO DESENVOLVIMENTO DE UM SOFTWARE DURANTE O PROCESSO DE IMPLANTAÇÃO DA MPS.BR

Belo Horizonte

2017

CARLA FLORIANA MARTINS

TRADIÇÃO E MODERNIDADE: DES (ENTENDIMENTOS) NO DESENVOLVIMENTO DE UM SOFTWARE DURANTE O PROCESSO DE IMPLANTAÇÃO DA MPS.BR

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia de Produção da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito parcial para obtenção do grau de Mestre em Engenharia de Produção.

Área de Concentração: Estudos Sociais do Trabalho, Tecnologia e Expertise

Orientador: Prof. Raoni Guerra Lucas Rajão

Belo Horizonte

2017

Martins, Carla Floriana

M383 t Tradição e modernidade: des (entendimentos) no desenvolvimento de um software durante o processo de implantação da MPS.BR. Belo Horizonte: UFMG, 2017.

143 f., il., enc.

Orientador: Prof. Raoni Guerra Lucas Rajão

Dissertação (Mestrado em Engenharia de Produção) – Universidade Federal de Minas Gerais.

Referência: f. 139-143.

1. Engenharia de Software. 2. Engenharia de Software. 3. Olhar sociotécnico. 4. Processo de Desenvolvimento de Software. 5. Comunidades de Prática. 6. Formas de Trabalho em Conjunto. I. Rajão, Raoni Lucas Guerra. II. Escola de Engenharia - UFMG. III Título.

CDU: 658.5

Ficha catalográfica elaborada por Emilce Maria Diniz – CRB- 6ª n. 1206

UNIVERSIDADE FEDERAL DE MINAS GERAIS

PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE PRODUÇÃO

UFMG

FOLHA DE APROVAÇÃO

TRADIÇÃO E MODERNIDADE: DES (ENTENDIMENTOS) NO DESENVOLVIMENTO DE UM SOFTWARE DURANTE O PROCESSO DE IMPLANTAÇÃO DA MPS.BR

CARLA FLORIANA MARTINS

Dissertação submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em ENGENHARIA DE PRODUÇÃO, como requisito para obtenção do grau de Mestre em ENGENHARIA DE PRODUÇÃO, área de concentração PESQUISA OPERACIONAL E ENGENHARIA DE MANUFATURA, linha de pesquisa Estudos Sociais da Tecnologia, Trabalho e Expertise.

Aprovada em 03 de maio de 2017, pela banca constituída pelos membros:

Prof(a). Raoni Guerra Lucas Rajão - Orientador
UFMG

Prof(a). Francisco de Paula Antunes Lima
UFMG

Prof(a). Henrique Luiz Cukierman
Universidade Federal do Rio de Janeiro

Belo Horizonte, 18 de maio de 2017.

AGRADECIMENTOS

O maior presente que podemos dar ou receber da vida é o conhecimento. É através dele que crescemos, evoluímos e podemos contribuir para um mundo mais honesto, justo e responsável. A ciência, a tecnologia e as formas como a sociedade se reconstrói a cada momento, fomentam a incansável busca pela informação e aprendizagem. Por isso, quando mais uma etapa de minha trajetória educacional é concluída, me sinto muito feliz e vitoriosa por ter a oportunidade para agradecer a algumas pessoas muito especiais que fizeram parte dessa historia.

À minha mãe e meu pai *in memorian* a quem a trajetória de vida não permitiu acesso à educação superior, mas que com muita sabedoria e valorização ao conhecimento e cultura sempre me incentivaram a buscar a educação de qualidade.

Aos meus irmãos, por ordem cronológica, Vladimir, Nádia, Lorena e Leonardo (e novamente a Leonardo, que muito me ajudou nos momentos de estudo e trocas de opiniões), e aos meus cunhados e irmãos do coração Erica, Wesley e Angelo. Meu muito obrigada por estarem sempre comigo e pelo amor incondicional que recebo de vocês. Amo muito vocês.

Meus sobrinhos, Vitor, Brenda, Emillie, Mariana e Maria Lucia, também por ordem cronológica, que amo tanto e que tanto vibram com minhas empreitadas e conquistas.

Minha madrinha, mãe e amiga Aparecida, que acompanhou comigo todo o processo me apoiando e incentivando.

Minhas amigas, irmãs e sócias Izabela, Amélia e Silvana (não contarei aqui se está em ordem cronológica) e a meu amigo e sócio Gustavo, pela amizade, parceria e ajuda irrestrita e incessante, participando comigo de todo o processo e me proporcionando a oportunidade de conciliar o trabalho com a escrita da dissertação, obrigada pela compreensão da minha ausência em momentos importantes de nossas vidas e da Praxis. Obrigada pelo apoio, amizade e carinho! A meu amigo Valmir!

Minha amiga, comadre, irmã e parceira, Jussara. Mãe da dupla dinâmica Lucas e David. Obrigada amiga por acreditar que eu conseguiria e por estar comigo em todos os altos e baixos dessa aventura. Você é muito importante na minha vida! Ao Lucas pela cartinha de incentivo na reta final, me deu alento e alegria!

Minha amiga, irmã e parceira Carla Adriani, que conhece cada vírgula desse trabalho e que não poupou esforços para me ajudar incluindo, aí, finais de semana e feriados. Agora acho que poderemos ter muitos finais de semana pra comemorar.

Ao meu orientador Raoni Guerra, por sua presença, carinho, atenção, competência, credibilidade e por ter acompanhado cada passo do desafio da conclusão de uma etapa tão significativa na minha vida. Obrigada Raoni!

À equipe do PPGEP, na presença, carinho e ajuda constantes dos professores Rodrigo Ribeiro e Francisco Lima e também ao Marcos e Karla, da secretaria de pós-graduação da PPGEP, meu muito obrigada!

Volto ainda para agradecer ao meu cunhado Wesley, meus irmãos Lorena, Leonardo e Vladimir, ao meu cumpadre amado e a meu primo Sergio. No meio da escrita houve um grande desafio, que eles assumiram pessoalmente, tornando mais leve minha jornada e minha trajetória!

Ao meu tio Tião. "O Cara"! E a Dona Lourdes e Vania, pela amizade e orações!

Por fim agradeço com muito carinho à equipe de colaboradores da Praxis, que estiveram presentes em cada momento dessa escrita. Sem a participação e colaboração de vocês a pesquisa não poderia ser realizada. Obrigada!

RESUMO

Este trabalho busca compreender algumas situações ocorridas durante a implantação do modelo MPS.Br (Modelo de Melhoria do Software Brasileiro) no contexto do desenvolvimento de um software. Buscou-se observar o impacto dessa implantação na forma como a empresa era operada anteriormente e como ela passou a operar a partir do uso dessa prática já conhecida no mercado brasileiro. A partir da pesquisa em campo, de inspiração etnográfica, foi possível perceber que transformações substanciais nas lógicas de produção, de resultado, de relação e de comunicação entre comunidades de práticas estabelecidas ocorreram dentro da empresa. Em particular, foi possível notar que a introdução do MPS.Br contribuiu para a perda do caráter mais artesanal que perdurava na empresa, transformando-a na direção de lógicas de origem industrial e instrumental. Além disso, após obter a certificação do MPS.Br as expectativas da empresa com relação ao aumento da produtividade em relação a quantidade de códigos escritos não foram as esperadas. A partir dessa reflexão, o presente estudo evidencia momentos em que prevaleceram as formas colaborativas de trabalho e questiona o uso de metodologias rígidas e tradicionais de desenvolvimento de softwares como as almejadas pelo MPS.Br e outras metodologias similares. A pesquisa ressalta, também, que um melhor entendimento sobre a atuação de Comunidades de Prática existentes num determinado contexto, e uma percepção sociotécnica das variáveis que envolvem o desenvolvimento de softwares podem contribuir com uma perspectiva para a Engenharia de Software mais sintonizada com os aspectos sociais e organizacionais das instituições. Desse modo, as organizações poderão oferecer condições que visem facilitar o trabalho colaborativo entre equipes e comunidades, obtendo a melhoria na produção de artefatos voltados para a ampliação de conhecimentos e conjunturas de transformações das organizações.

Palavras-chave: Engenharia de Produção. Engenharia de Software. Olhar sociotécnico. Processo de Desenvolvimento de Software. Comunidades de Prática. Formas de Trabalho em Conjunto.

ABSTRACT

This work seeks to understand the contradictions related to the implementation of the MPS.Br model (Brazilian Software Improvement Model) deployed in the context of the development of a software. This study observed the impact of the MPS.br implementation in the way the company operated. Based on ethnographic-inspired field research, it was possible to perceive that substantial changes in the logics of production, results, relationships and communications between the communities of practices within the company. In particular, it was possible to note that the introduction of the MPS.Br contributed to the loss of the more artisanal, collaborative and communicative character that persisted in the company, transforming it towards the establishment of industrial and instrumental logics. In addition, after obtaining MPS.Br certification, the company's expectations to increase productivity were frustrated. Instead, software development became slower and more conflictive, compromising product quality and business viability. Based on this, the present study highlights the importance of collaborative forms of work and questions the use of rigid and traditional methodologies of software development such as MPS.Br. In addition, the research emphasizes that a better understanding of the performance of existing Communities of Practice in a given context, and the perception of the sociotechnical aspects of development of software can contribute to a perspective for Software Engineering more productive and less conflictual. In this way organizations can offer conditions that aim to facilitate collaborative work between teams and communities, obtaining, in this way, an improvement in the production of artifacts geared to the expansion of knowledge and conjunctures of organizations' transformations.

Keywords: Production Engineering. Software Engineering. Sociotechnical look. Software Development Process.Communities of Practice. Ways of Working Together

LISTA DE FIGURAS E QUADROS

LISTA DE ABREVIATURAS

CMM	CAPABILITY MATURITY MODEL
CMMI	CAPABILITY MATURITY MODEL INTEGRATION
DS	DESENVOLVIMENTO DE SOFTWARE
ES	ENGENHARIA DE SOFTWARE
FINEP	FINANCIADORA DE ESTUDOS E PROJETOS
GP	GERENTE DE PROJETOS
GPR	GERENCIAMENTO DE PROJETO
GRE	GERENCIAMENTO DE REQUISITOS
MPS.Br	MODELO DE MELHORIA DO PRODUTO DE SOFTWARE BRASILEIRO
SEI	SOFTWARE ENGINEERING INSTITUTE
SOFTEX	ASSOCIAÇÃO PARA PROMOÇÃO DA EXCELÊNCIA DO SOFTWARE BRASILEIRO

SUMÁRIO

1 INTRODUÇÃO

O Desenvolvimento de Softwares (DS) é um processo complexo, que se apoia em pilares que vão desde as competências técnicas necessárias à função de um desenvolvedor, até as habilidades relacionais que podem acontecer entre eles e outros grupos envolvidos no processo (FRANÇA E SILVA, p. 25, 2007). Não é novidade para a literatura sobre Desenvolvimento de Software que empresas em várias partes do mundo enfrentam altos índices de falhas, com apenas 39% dos projetos considerados com sucesso, 43% que sofrem mudanças ou são realizados acima do tempo e orçamento e 18% analisados como fracassados na sua totalidade (CHAOS MANIFESTO, 2013).

Por esse motivo as empresas desse setor buscam estabelecer estruturas e processos de DS com alto nível de controle de modo a obter um modelo de produção mais "maduro" e melhorar a qualidade de seus produtos. Foi a partir desse contexto que a Empresa (foco do estudo de caso apresentado nessa dissertação) após consulta direta à equipe de desenvolvimento e, também a partir de pesquisa de mercado, optou por implementar em sua estrutura de funcionamento o MPS.Br (Modelo de Melhoria do Produto de Software Brasileiro), no chamado nível G de maturidade.

Este processo decisório ocorreu concomitantemente à resolução pela escrita e lançamento de um novo produto: um Software de Gerenciamento de Bibliotecas acoplado a uma Rede Social de Leitores. Software este que deveria vir com características modernas e inovadoras para o mercado que o receberia e ainda ser imbuído de novos conceitos e novas construções cognitivas surgidas das tendências contemporâneas de interatividade e conectividade.

O corpo técnico, ao sugerir o modelo MPS.Br, tinha como pretensão envolver mais a alta gestão nos processos ocorridos dentro do setor DS e limitar as intervenções e mudanças de projeto durante o percurso de codificação. Já a empresa aceitou a proposta de padronização porque pretendia oferecer um novo produto ao mercado devidamente documentado, com rastreabilidade de códigos,

em tempo delimitado além de orçamento e qualidade prenunciados pelo modelo de maturidade. Essa expectativa provém da promessa dos promotores de modelos como o MPS.Br que afirmam que o DS de sucesso está necessariamente ligado à conformidade com modelos preestabelecidos (JERÔNIMO JUNIOR; CARVALHO, 2012), com especificações vindas de especialistas de domínio, num tempo determinado de entrega do produto e dentro de um orçamento prévio (PRESSMAN, 2011, p. 78).

No entanto o desenrolar dessa história se apresentou de forma diferente do que se esperava e um novo cenário se configurou durante o desenvolvimento do sistema. Uma vez implementada a metodologia, tornaram-se mais comuns situações de conflito envolvendo todos os setores da empresa, desde a gestão, perpassando pelo desenvolvimento, setor comercial e financeiro. Foi nesse contexto que a pesquisadora, na qualidade de mestranda, mas também de sócia e especialista de domínio na referida Empresa, foi obrigada a rever alguns de seus conceitos, principalmente no que se refere a metodologias de DS.

Nessa busca, os estudos sociotécnicos sobre o DS ofereceram uma fonte importante de reflexão. Esses estudos trazem uma visão mais atenta às circunstâncias organizacionais, culturais e históricas dentro da qual ocorre o desenvolvimento de software e, a partir daí apresentam perspectivas que vão além das prescrições encontradas nos manuais de engenharia. Além disso, esses estudos enfatizam que a escuta e o diálogo entre pares e demais envolvidos no processo de DS deve ser problematizado e percebido como o caminho mais acertado a se seguir (CUKIERMAN; TEIXEIRA; PRIKLANDNICKI, 2007; LEAL, 2008; ROBILLARD, 1999).

Vinculados aos estudos sociotécnicos, também foi relevante para essa pesquisa a literatura sobre limites e formas de trabalho conjunto, tais como Coerção, Coordenação, Cooperação e Colaboração e sua influência para a fruição de ambientes idealizados de inovação (WENGER; LAVE, 1991, ENGESTRON, 1991, RAJÃO, 2011).

Desta forma, o objetivo do presente trabalho é compreender as dinâmicas sociais em torno do Desenvolvimento de Software. Em particular, compreender a dinâmica da proposta de implantação do modelo MPS.Br (Modelo de Melhoria do Software Brasileiro) realizada no contexto do desenvolvimento de um Software de Bibliotecas em uma empresa de desenvolvimento de softwares, sediada em Belo Horizonte/MG. Buscou-se observar, nesse estudo, o impacto dessa implantação na forma como a empresa era operada anteriormente e como ela passou a operar a partir do próprio processo de implantação e uso desse modelo, já habitual no mercado brasileiro. Para tanto, este estudo adota a perspectiva a partir de uma visão sociotécnica e adentra nos estudos sobre comunidades de prática e diferentes tipos de trabalho em conjunto que ocorrem ao longo do processo (WENGER, 1991; 2006; 2007, RAJÃO, 2011).

Essa é uma pesquisa de influência etnográfica onde reforçamos o caráter social do estudo realizado. O objetivo conceitual dessa metodologia é, a partir de observações e imersões em campo, ajudar o pesquisador a trazer à luz como se dão as relações entre um grupo social e suas tarefas dentro de um espaço organizacional previamente escolhida. A etnografia como metodologia, nos oportuniza com critérios de acompanhamento das ações desenvolvidas durante uma tarefa ou trabalho específico, observar e trazer à tona modos de fazer do trabalhador, que nem sempre seriam captadas em técnicas tradicionais.

Para tanto, durante a imersão em campo são utilizadas como formas de registro: vídeos, entrevistas, gravações, autoconfrontações etc. Uma das críticas a essa linha de pensamento, é que, por se tratar de metodologia que envolve pessoas, não é possível garantir o grau de comprometimento do observado com a pesquisa, e, consequentemente, com as respostas ou reações perante entrevistas, gravações e mesmo autoconfrontações.

Por isso é importante que o pesquisador trave uma relação de respeito e confiança com o entrevistado para se ter o máximo de comprometimento do participante com o processo. Os espaços conceituais para essas análises serão os estudos de comunidade de prática aliados às formas de trabalho em conjunto,

como suporte para o entendimento de como a equipe se comporta, como as relações se dão e como podemos entender estas variações.

A posição da pesquisadora na empresa é a de especialista de domínio, alocada para trabalhar juntamente com uma equipe multidisciplinar de analistas e desenvolvedores. A pesquisadora também atuou com equipe de implantação do modelo MPS.Br na empresa e, após a certificação no modelo, adquiriu cotas societárias na organização. Essa posição também interferiu nas tensões uma vez que a pesquisadora deveria fazer o escopo dentro do processo ao mesmo tempo em que o implantava.

Todos da empresa vinham acompanhando o trabalho e sempre foram muito disponíveis para fazerem as entrevistas. A pesquisadora teve muita abertura por parte da Gestora e da equipe do DS. Os Desenvolvedores foram envolvidos nessa pesquisa de outros modos também. Através do envio de materiais e da formação de um grupo de trabalho que fez leituras críticas de textos sobre a perspectiva sociotécnica sobre a ES. Inclusive mudando um pouco a linguagem da equipe e da empresa como um todo que busca não dividir mais os aspetos técnicos dos não técnicos do Software.

Trataremos os aspectos reais da construção do software já num âmbito de seus usos locais, entendendo que algumas tensões geradas durante o processo podem ser espelho de dificuldades comunicacionais existentes entre profissionais de áreas diferentes que participam do Desenvolvimento de um software feito paralelamente à implantação do modelo MPS.Br, entre outras situações.

A presente dissertação de mestrado foi dividida da seguinte forma. O capítulo seguinte apresenta uma revisão de literatura para que seja possível compreender as diferentes linhas de pensamento sobre como deve ser realizado e compreendido o desenvolvimento de software. O capítulo 3 trará alguns elementos teóricos que serão relevantes para análise dos dados, entre eles os conceitos de Alienação do Trabalho, Razão Comunicativa, Formas de Trabalho em Conjunto e Comunidades de Práticas. O capítulo 4 traz mais detalhes do desenvolvimento do software.

O capítulo 5 apresenta o contexto histórico e organizacional onde se insere o processo de implementação do MPS.Br e algumas mudanças organizacionais e conflitos observados. O capítulo 6 trata do momento mais tenso vivenciado pela empresa, onde os Desenvolvedores reconhecem as dificuldades do modelo MPS.Br garantir as entregas dos resultados prometidos e a busca do grupo por soluções. E, o capítulo 7, finaliza o trabalho trazendo uma discussão que dialoga com a literatura e a abordagem teórica, além de considerações e contribuições práticas trazidas pela pesquisa.

2 DAS PROMESSAS DA ENGENHARIA DE SOFTWARE AOS ESTUDOS SOCIOTÉCNICOS

A Engenharia de Software (ES) é uma disciplina que reivindica a possibilidade de, através da utilização de modelos, metodologias, ferramentas e processos, melhorar o DS visando à criação de softwares cada vez mais adequados e aderentes às propostas e necessidades de seus demandantes (HIRAMA, 2012; SCHACH, 2009; SOMMERVILLE, 2008; REZENDE, 2005).

Porém, muito se tem discutido acerca dos verdadeiros benefícios do uso dessas metodologias quando nos deparamos com situações ou demandas reais de trabalho (SILVA; LIMA, 2005). Apesar dessas promessas da ES, estudos demonstram que a análise do contexto de organização do trabalho podem congregar esforços para resolver lacunas de concepção de softwares percebidas, geralmente, ao final dos processos tradicionais de DS (LEAL, 2008; FERREIRA; LIMA, 2005; TEIXEIRA, 2007; SILVA; LIMA, 2005).

Preocupados com essas lacunas, entendemos como muito importante nos debruçarmos nos recentes estudos que propõem um uso de recursos mais holísticos, sociais e transversais como soluções viáveis para velhos questionamentos e problemas que trazem tanto prejuízos financeiros como emocionais às instituições e a seus colaboradores ao longo do tempo.

Para fins de compreensão mais orgânica sobre a busca de soluções que possam amparar grupos que se preocupam em diminuir o quanto possível consequências inesperadas para esse tipo tradicional de solução, seguiremos na literatura que, a partir das Ciências Sociais, busca compreender determinadas situações observadas no campo empírico.

Começaremos com uma curta explanação sobre a gênese dos estudos de Desenvolvimento de Softwares e Engenharia de Software, os motivos que levaram os profissionais da área em criar metodologias para dar suporte às ambiências de DS, um histórico de alguns de seus modelos tradicionais e sua

evolução, até chegarmos às atuais propostas sociotécnicas para melhoria do Desenvolvimento de Software, que entendemos como necessárias e importantes para um resultado mais condizente com a expectativa de uma sociedade cada vez mais dinâmica em mutação.

2.1 O nascimento da ES e a primeira "crise do software"

Em meados da década de 60, devido à massificação e demandas cada vez mais crescentes por produtos de software, os problemas, falhas e prejuízos relativos aos processos de DS se tornaram muito constantes. Em uma convenção patrocinada pela NATO/OTAN (North Atlantic Treaty Organization) aconteceu a 1ª Conferência de Engenharia de Software, sediada na Alemanha, em 1968.

Neste congresso foi empregado, pela primeira vez, o termo "Crise do software" (CERUZZI, 2006, p. 128). Naquele momento, os participantes do evento discutiram que havia uma "crise" (atrasos nas entregas de sistemas, os custos de produção de software maiores que os previstos, pouca confiabilidade em relação aos produtos entregues, entre outros) a qual deveria ser amplamente discutida, não somente a nível técnico, mas também a nível político e estratégico devido à gravidade do problema.

Também foi reconhecido, nesse mesmo evento que "essa crise de produção de software", não acabaria rapidamente, "mas que poderia, e deveria ser gerida" da melhor forma possível (ibidem). Além disso, o evento usou, provocativamente, o termo "Engenharia de Software", até então pouco conhecido, dando a entender que faltava aos programadores "fundamentos teóricos e as disciplinas da prática cotidiana que se encontram nos campos da Engenharia Tradicional" (idem, p. 129).

Ao longo dos anos, a área de DS foi se personalizando e se tornando mais consolidada e estruturada. Porém, esses avanços não foram suficientes para que os problemas levantados (tempo, preço, qualidade), fossem resolvidos de forma satisfatória, principalmente para as organizações, que continuavam investindo

em demandas otimistas, mas ainda com muitas incertezas em relação ao resultado final de suas iniciativas.

De qualquer forma, o entendimento que DS traz em si um alto grau de complexidade e a noção percebida, já à época, de que havia pouquíssimas metodologias ou técnicas que auxiliassem o desenvolvedor em seu trabalho, (ROCHA; MALDONADO; WEBER, 2001) fez com que as discussões continuassem. O objetivo era encontrar soluções que minimizassem os problemas detectados e legitimasse o fazer técnico do desenvolvedor validando, de alguma forma, seu trabalho.

Sendo assim, para solucionar a "Crise do Software" a nascente Engenharia de Software sugere que

> Apenas a adoção de processos efetivamente industriais para produção de software poderá fazer essa área desenvolver-se mais rapidamente, com mais qualidade e, finalmente, sair dessa dificuldade crônica que já nem pode ser chamada de crise (WAZLAWICK, 2013).

De acordo com Wazlawick (2013), a Engenharia de Software, "é o processo de criar, estudar e otimizar os processos de trabalho para o desenvolvedor do software". Colocado dessa forma, o autor assinala que o desenvolvedor é o executor do DS e, o engenheiro é o estudioso e provedor dos processos métodos e ferramentas necessárias para o desenvolvimento do software bem como o responsável por acompanhar tal processo.

Abaixo colocaremos os principais conceitos da ES tradicional, a começar pela definição de alguns papéis clássicos do DS: os atores, os documentos, e os processos.

2.2 Os atores do Desenvolvimento de Software

Desde os estudos da década de 60 até os dias atuais, muito se assinala sobre a importância de as funções desenvolvidas pelos atores do DS serem demarcadas conceitualmente para melhor acompanhamento de execução das tarefas. Dentro

dessa definição, as funções clássicas de capturar os requisitos e executá-los são de competência do desenvolvedor/programador, independente do papel que ele ocupa no processo de DS.

Alertando o leitor que podem existir várias colocações ligadas ao processo de DS, Wazlawick (2013) subdivide alguns desses papéis e suas funções as quais mencionaremos *grosso modo*, os considerados como mais conhecidos ou comuns nas instituições:

a. *Engenheiro de Software*: Ele participa do planejamento e acompanhamento da evolução dos projetos de toda empresa. Avalia e identifica problemas e projeta mudanças caso necessária. É também responsável pela reavaliação dos projetos;

b. *Gerente de Projetos*: cuida de um ou mais projetos específicos. Verifica se o processo desenhado pelo Engenheiro de Software está sendo executado nos prazos e orçamentos previstos;

c. *Especialista/Analista de domínio:* responsável pelo levantamento dos requisitos e sua modelagem. Ele deve entender a demanda do cliente em relação ao sistema a ser desenvolvido;

d. *Designer:* a partir das especificações do analista propõe tecnologias para executar o projeto.

e. *Desenvolvedor/Programador:* Ele constrói a solução física a partir das especificações do *designer*. Deve ter expertise em linguagens e ambientes de programação. É recomendável que tenha conhecimentos sobre testes e depuração de software.

Com essas demarcações em seu escopo tradicional, a Engenharia de Software ressalta a importância de definir funções e responsabilidades, de modo rígido, para garantir, na visão da mesma, o sucesso dos processos propostos (ENGHOLM JR., 2010, p. 67; SCHACH, 2010, p. 271; LOBO, 2009, p. 115).

Dito isso, é importante ressaltar que estruturas funcionais assim definidas raramente são possíveis segundo Wazlawick (2013) e que em muitos casos, principalmente para as pequenas e médias empresas, os papéis podem se misturar durante a atuação de uma mesma pessoa ou equipe. Nosso olhar durante o trabalho empírico esteve voltado para todas as funções exercidas, porém, com foco no desenvolvedor e em suas reações e percepções perante o processo de DS o qual estavam envolvidos.

2.2.1 Os documentos

Requisitos fazem parte do universo do Desenvolvimento de Softwares. De acordo com Nardi e Falbo (2006, p. 7) "requisitos são sentenças que descrevem serviços que um sistema deve prover restrições que ele deve obedecer e características que ele deve possuir".

Além disso, na maior parte das vezes, requisitos são definidos a partir do escopo de um projeto (PRESSMAN, 2007; BRAY, 2002). Entendendo aqui como "projeto", o conceito dado por Pressman, (2007, p. 206) como "um conjunto de atividades que englobam conceitos, princípios e práticas que levarão ao desenvolvimento de um sistema ou um produto com alta qualidade".

Também em proposta clássica, Pressman (2011) trata dos requisitos no âmbito da Engenharia de Software, o que acontece no momento em que os interessados em construir um sistema se reúnem com o Engenheiro de Software para definir o escopo inicial do projeto. O autor vê esse momento como uma tarefa a ser conduzida com o objetivo de se formar uma "base sólida" para o planejamento do projeto de escrita de um software e sua execução.
Neste sentido, requisitos "brotam" desde as primeiras discussões sobre a modelagem do software e vão se expandindo à medida que informações tais como restrições, funções, objetivos, principais características do sistema, são levantadas pelos grupos envolvidos. A partir daí e de acordo com os dados coletados pelos analistas, eles já podem passar por refinamentos e extensões até

estarem aprovados e validados para o início do Desenvolvimento do Sistema (PRESSMAN, 2011, p. 126).

Ainda em relação a requisitos, Martins (2010, p. 181), defensor das metodologias ágeis de DS, acrescenta que um processo de "captura de requisitos" é importante por envolver os interessados na construção do sistema e na produção e escrita das especificações que serão necessárias para o Desenvolvimento do Software.

Proveniente de uma área mais técnica da Engenharia de Software, Martins (2010, p. 181) sintetiza que requisitos são "características funcionais e não funcionais que o sistema precisa apresentar". E conceitua os requisitos chamados por ele de funcionais como os que definirão o comportamento do sistema, capturado por meio de casos de uso, (estudo de uso do sistema), "que documentam as entradas, os processos e as saídas geradas".

O autor completa seus posicionamentos em relação a requisitos afirmando que "os requisitos não funcionais são compostos por características não necessariamente associadas ao comportamento do software" (Entradas e Saídas), tais como: Usabilidade, Confiabilidade, Performance e Suporte. No texto especifica cada componente:

a. *Usabilidade:* facilidade de utilização pelas pessoas, aspectos visuais, documentação e material de treinamento;

b. *Confiabilidade*: Robustez, proteção contra falhas, recuperação em caso de erro e precisão;

c. *Performance:* O sistema deve apresentar disponibilidade, tempo aceitável de execução de transações, bom tempo de resposta, baixo uso de memória e outros recursos computacionais;

d. *Suporte:* esses requisitos estão associados à facilidade de se manter o sistema em produção, definindo plataforma, linguagem, ambiente de desenvolvimento e testes, dentre outros.

É interessante reforçar que a maneira como a ES tradicional sugere que criar requisitos completos e robustos seja condição necessária e suficiente para se desenvolver um software com sucesso. Isso, pressupondo que requisitos podem ser feitos de modo objetivo e compreendidos de forma inequívoca por todos os atores envolvidos.

Essa mesma linha tradicional, irá defender que, através de metodologias e processos, também rígidos, os resultados esperados a partir da escrita correta dos requisitos serão alcançados mais facilmente, como veremos abaixo em relação aos processos. (HIRAMA, 2012; SCHACH, 2010; SOMMERVILLE, 2008; REZENDE, 2005).

2.2.2 Os processos

Uma vez definidos os documentos, os processos passam a ser a ponte, dentro das propostas tradicionais, para um bom DS. Nessa perspectiva, o "processo é a parte do desenvolvimento de software que oferecerá uma metodologia para a prática da ES" (PRESSMAN, 2011, p. 52). Talvez por esse motivo, a Engenharia de Software seja a área do conhecimento que mais se utiliza dos estudos, propostas e conceitos para aplicações em Desenvolvimento de Software.

De acordo com Leal (2008, p. 18), é da natureza da ES, a partir de seu histórico ocidental racional buscar, o máximo possível, processos que garantam a qualidade de um produto de software, entendendo este como "uma sequência de instruções escritas para serem interpretadas por um computador com o objetivo de executar tarefas específicas".

Assim, desde uma visão mercadológica que o equilíbrio financeiro das empresas está ligado à produção e entrega dos artefatos dentro de tempo e orçamentos previamente determinados, que executivos da área de tecnologia da informação, dentre outros, passaram a buscar procedimentos de controle e documentação.

Esses executivos tinham como objetivo garantir o entendimento e continuidade dos processos pelos Desenvolvedores. Um pouco disso pensando na rotatividade de pessoas, uma vez que "precisavam reduzir sua dependência das habilidades desses profissionais, sendo necessárias metodologias efetivas para o gerenciamento dos projetos de software e controle do processo de desenvolvimento" (FERNANDES; TEIXEIRA, 2007, p. 23).

Neste sentido, talvez a expectativa em se usar modelos e processos clássicos para DS seja a crença na possibilidade de se levantar atividades metodológicas genéricas que poderiam ser identificadas em todos os projetos, o que levaria os empreendedores, instituições e engenheiros de software a serem capazes de trabalhar a partir de padrões, entendidos neste estudo, como padrões ou modelos tradicionais.

2.2.2.1 Modelo cascata

O Modelo Cascata (ou *Waterfall Model*) é um modelo proposto na década de 70 por Winston Royce (PRIKLADNICKI; WILLI; MILANI, 2014, p. 232; PRESMAN, 2011, p. 59; HIRAMA, 2012, p. 2) e é ainda muito utilizado para o Desenvolvimento de Software.

De acordo com Hirama, (2012) o modelo cascata é "um processo focado nos documentos e artefatos", (p. 38) e "define um desenvolvimento linear e sequencial do software" (p. 31). O autor acrescenta que "da maneira como é proposto, ele não se adapta facilmente às novas exigências do mercado" (p. 31) e faz essa ponderação devido ao foco do modelo cascata almejar, ao fim de seu processo, entregar um sistema completo.

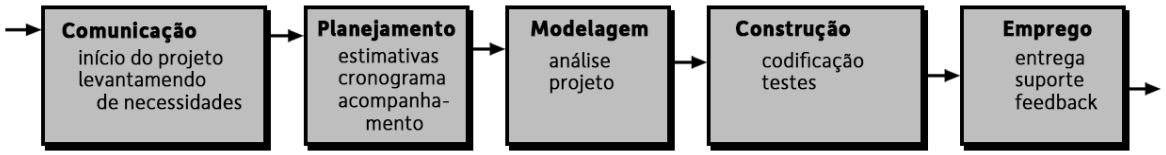
Essa rigidez acaba por trazer dificuldades na "acomodação de mudanças depois que o projeto está em andamento" segundo Hirama (2012, p. 28, p. 31). Em processos maiores e com elementos pouco mutáveis, tais como projetos militares que lidam com informações consolidadas, essa metodologia já gera problemas por causa das expectativas e realidade de entrega dos produtos finais.

Para minimizar esses problemas Larman (2007, p. 51) afirma que em sociedades cada vez mais dinâmicas modelos mais iterativos trariam a possibilidade de se absorver essa perspectiva do mutável a partir de entregas intermediárias,

Conhecido também como *Ciclo de Vida Clássico*, Pressman (2011, p. 59), o modelo em cascata é representado graficamente, na maioria dos casos, como um modelo linear e sequencial. No entanto, autores têm sinalizado a intencionalidade iterativa de Royce ao desenvolver o modelo, segundo eles, mal compreendido pelos Desenvolvedores de Softwares que fizeram a leitura do modelo como linear apenas (PRIKLADNICKI; WILLI; MILANI, 2014, SCHACH, 2010, p. 39; PRESSMAN, 2011, p. 59).

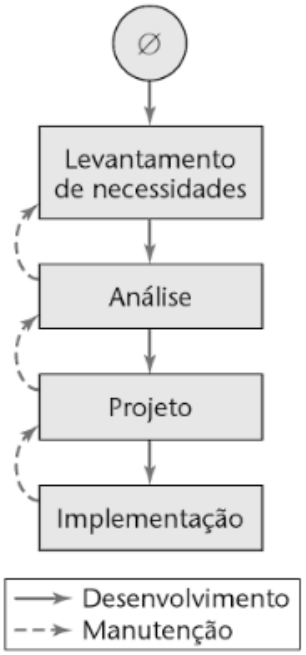
Desta forma, Pressman (2011, p. 59), enfatiza que Royce previa os "feedbacks loops" ressaltando que as organizações, em sua maioria, aplicam o método "como se fossem estritamente lineares". Para fins gráficos, traremos um modelo representado por Pressman (2011, p. 60), FIG. 1, e outro, simplificado, que prevê uma possibilidade de se retroceder no processo sugerido por Royce, FIG. 2.

FIGURA 1 - MODELO DE CICLO DE VIDA CLÁSSICO - CASCATA



Fonte: Pressman (2011, p. 60).

FIGURA 2 - VERSÃO SIMPLIFICADA DO MODELO DE CICLO DE VIDA EM CASCATA COM INTERAÇÕES



Fonte: SCHACH (2010, p. 39).

Vale lembrar que, de acordo com Schach (2010, p. 36), o modelo de ciclo de vida linear e sequencial, sem necessidade de correções ao longo do processo, é considerado o padrão ideal pelos usuários deste modelo, o que demonstra o desejo da área em controlar o processo a partir do levantamento completo das necessidades do demandante do projeto.

2.2.2.2 Fábrica de software

De modo geral, os grupos e empresas que buscam as soluções tradicionais e formais que levam ao tão almejado sucesso no processo de DS, tentam operar numa estrutura conhecida como “Fábrica de Software”.

De acordo com Castor (2006), este termo foi usado pela primeira vez nos anos 60 por Bob Bremer no texto "*Position Paper for the Panel Discussion on the Economics of Program Production"* onde ele defende que o controle dos

processos deve ser da empresa e não somente estar nas mãos do "trabalhador". Mais tarde, Mcllorol (1969 apud Castor 2006), Bauer (1971 apud Castor 2006), também utilizaram o termo em textos, intercalando, a isso, o caso das empresas Hitachi, no ao de 1969, *System Development Corporation* em 1975 e dos governos do Japão e dos Estados Unidos em 1976 e 1977, que utilizaram o termo Fábrica de Software para designar que usavam metodologias de controle específicas em seus ambientes de produção e desenvolvimento.

A partir da década de 80 até os dias atuais, o termo foi usado por empresas e governos e difundido para o mundo. De acordo com Fagundes, (2004), fábrica de software pode ser definida

> Como uma estrutura de desenvolvimento que reúne alguns requisitos básicos, como a capacidade de atender clientes, de projetar o produto, de planejar e controlar a produção, de produzir e de controlar a qualidade.

Já Fernandes e Teixeira, (2007), afirmam que a Fábrica de Software é um negócio e deve gerar produtos de software, de acordo com as especificações devidamente documentadas pelos usuários ou cliente *"da forma mais produtiva e econômica possível"*.

De qualquer forma, o termo cunhado como Fábrica de Software expressa o entendimento que modelos industriais seriam capazes de minimizar e dar qualidade aos softwares desenvolvidos se fossem produzidos nessa perspectiva.

Porém, esse juízo vem sendo questionado por autores que destacam a necessidade de abordagens mais sociais (políticas, dialógicas, culturais, organizacionais, entre outras), de forma que os requisitos técnicos e não técnicos possam ser tratados de preferência, de forma conjunta e indissociável (TEIXEIRA, 2007; LEAL, 2008; CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007).

Prova que as abordagens sociais podem ser mais válidas e atenderem à contemporaneidade é que mesmo usando metodologias e processos organizados

em forma de Fábrica de Software, o DS continuou a evidenciar falhas que impactavam na economia e qualidade de seus produtos.

Mesmo assim, e ainda nos moldes tradicionais, estudos continuam sendo feitos a partir do paradigma de controle, documentação e formas de atender às empresas e demais organizações desenvolvedoras de softwares dentro de modelos pré-estabelecidos. Foi a partir da continuidade deste movimento formal que vieram os modelos de maturidade para o Desenvolvimento de Softwares e suas certificações, tais como o CMM, CMMI e o próprio MPS.Br, conforme veremos abaixo.

2.2.2.3 CMM, CMMI

Em sua gênese, os estudos de modelo de maturidade, estão ligados às primeiras pesquisas sobre melhorias de processos e se consolidaram a partir da década de 80 quando aplicados por Watts Humphrey na IBM (International Business Machines). Ali, as premissas dos então criados CMMs (Capability Model Maturity), já na década de 80 foram utilizadas como modelo para se melhorar os processos em organizações (*CMMI® for Development, Version 1.3).*

Esse caminhar perpassa por grandes organizações, tal como o SEI (Software Engineering Institute), vinculado à universidade norte-americana Carnegie Mellon em Pittsburgh, criado para atender a uma demanda governamental de organização de processos para desenvolvimento de software. O SEI instituiu o conceito de CMM (Capability Model Maturity) e iniciou, a partir daí o desenho de modelos básicos para gerenciamento de softwares em organizações, publicados, inclusive em um livro editado em 1995, o Capability Maturity Model: Diretrizes para Melhorar o processo de software.

Um modelo de maturidade, de acordo com o documento *CMMI® for Development, Version 1.3: Improving processes for developing better products and services*, pode ser entendido como um conjunto organizado de elementos necessários para avaliar alguns aspectos da maturidade de uma organização e

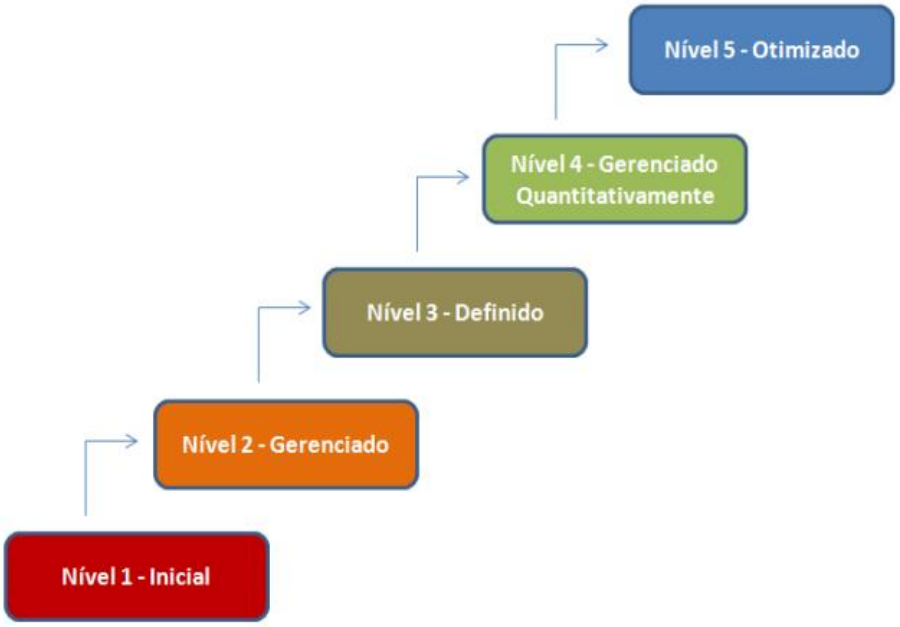
também trazer elementos para a melhoria de processos visando aumento da maturidade da organização ao longo do tempo. O CMM é considerado, de acordo com o documento, como uma "representação simplificada do mundo".

O CMMI (Capability Maturity Model Integration) "veio como uma das evoluções do CMM CHRISSIS (2007) e também dos modelos: SW-CMM - Capability Maturity Model for Software; SECM - System Engineering Capability Model, e IPD-CMM - Integrated Product Development CMM" (MCT, 2007). Seu enfoque está no desenvolvimento de softwares com maior qualidade e com menores incidências de erros em sua confecção. Em tal modelo, a maturidade da organização está arranjada por níveis, que vão do 1 ao 5 e possui duas formas de representações, por estágio ou contínua. Pouquíssimas empresas no mundo que utilizam o modelo CMMI para desenvolvimento de softwares possui o nível 5, o otimizado.

O objetivo do uso do modelo era o de melhorar os processos das organizações e melhorar suas habilidades e competências para lidar com a gestão do desenvolvimento, aquisição e manutenção de produtos e serviços. A ideia seria manter as práticas que já são consideradas consolidadas na instituição dentro de uma estrutura organizativa para que fiquem visíveis e facilitem à organização estabelecer prioridades de melhoria.

Os níveis e suas denominações, grosso modo, podem ser representados segundo o seguinte organograma:

FIGURA 3 - NÍVEIS DE MATURIDADE CMMI



Fonte: Site oficial CMMI.

Para se pleitear uma certificação CMMI é necessária uma avaliação por parte de um avaliador credenciado pelo SEI, que vai definir em qual nível de maturidade a empresa ou organização está e qual nível de maturidade passará a ser a meta da organização.

O processo delineia as competências necessárias para que a empresa chegue ao nível de maturidade almejado e passe a buscar os meios para adquirir essas competências. No Brasil, temos estatísticas da implementação do modelo CMM desde 1997 e de sua evolução, o CMMI (Capability Maturity Model Integration) a partir de 2003.

2.2.2.4 MPS.Br

O MPS.Br (Melhoria de Processos do Software Brasileiro), baseou-se no modelo CMMI, nas normas ISO/IEC 12207 e ISO/IEC 15504 e outras normas pertinentes de forma adaptada ao mercado brasileiro. Nascido para atender à demanda brasileira de observação a critérios de desenvolvimento de software, o Programa, de acordo com seu site oficial, foi idealizado para acolher, principalmente, às pequenas e médias empresas de desenvolvimento de software do país devido ao custo reduzido da certificação (SOFTEX, 2012).

Este projeto/modelo tem apoio do Ministério da Ciência e Tecnologia, do FINEP (Financiadora de Estudos e Projetos) e do Banco Interamericano de Desenvolvimento envolvendo no Brasil, a Associação para Promoção da Excelência do Software Brasileiro (SOFTEX), o governo e algumas universidades e tendo como um dos objetivos do projeto reproduzir o modelo MPS na América Latina. A finalidade do programa MPS.Br é o aumento da competitividade das organizações pela melhoria de seus processos de desenvolvimento de softwares.

Uma vez que o modelo é inspirado nas bases conceituais do CMMI-DEV, ele também prevê níveis de maturidade como metas para as organizações, que vão do G ao A, totalizando sete níveis de maturidade: G (Parcialmente Gerenciado), F (Gerenciado), E (Parcialmente Definido), D (Largamente Definido), C (Definido), B (Gerenciado Quantitativamente) e A (Em Otimização).

Tal modelo também se concretiza em forma de certificações a partir desses níveis. “O MR-MPS-SW baseia-se nos conceitos de maturidade e capacidade de processo para a avaliação e melhoria da qualidade e produtividade de software e serviços correlatos e também para a melhoria da qualidade e produtividade dos serviços prestados”.

Dentro deste contexto, ele possui quatro componentes: Modelo de Referência MPS para Software (MR-MPS-SW), Modelo de Referência MPS para Serviços

(MR-MPS-SV), Método de Avaliação (MA-MPS) e Modelo de Negócio para Melhoria de Processo de Software e Serviços" (MPS.Br - melhoria de processo do software brasileiro: guia geral MPS de software (SOFTEX, 2016, p. 6).

Assim como no caso da decisão pela certificação CMMI, a certificação MPS.Br exige passos que vão desde a opção por agregação da organização a empresas certificadoras até a visita e acompanhamento de avaliadores certificados pela SOFTEX (organização responsável no Brasil pela compilação, atualizações do modelo, formação de certificadores e chancelamento de credenciados).

Ao final do processo, os avaliadores verificarão se a organização ou empresa, ao seguir o modelo e requisitos pertinentes a cada nível, está apta a receber a certificação e em qual nível de maturidade. É um movimento que vem crescendo no Brasil e, de acordo com a SOFTEX, as empresas e organizações que buscam a certificação MPS.Br vêm crescendo ao longo dos anos, superando, em quantidade, às certificações CMMI, o que reforça que, mesmo com evidências que as falhas e problemas de processo continuam a ocorrer, as empresas ainda buscam modelos industriais e tradicionais de controle para tentar resolver seus problemas de qualidade e DS.

E nesse contexto, as metodologias ágeis vieram tentar soluções que diminuíssem as falhas ou que pelo menos as mesmas pudessem ser percebidas mais próximas ao tempo de construção do sistema. No próximo tópico trataremos dessa nova fase da ES, e também de algumas controvérsias observadas em seu percurso.

2.3 A segunda crise do software e as metodologias ágeis

As Metodologias Ágeis foram pensadas com o ideal de transpor as fraquezas conhecidas e percebidas da Engenharia de Software, oferecendo técnicas a serem seguidas para se chegar de forma mais rápida e assertiva à construção do software conforme Pressman (2011, p. 82). Elas vieram como reação ao enrijecimento do Desenvolvimento de Software proposto pela ES.

Surgiram em um momento que a ES tradicional percebeu que processos amarrados em modelos de registro e controle muito documentados não garantiram entregas nos prazos e nem o desenvolvimento de produtos sem falhas.

Assim, dentro de um cenário de situação econômica mundial e mercado consumidor cada dia mais dinâmico, um grupo desenvolveu e divulgou a *Agile Alliance*, uma organização sem fins lucrativos de adesão global, que se empenha em "fazer avançar os princípios e práticas de desenvolvimento ágil apoiando os grupos que desejem explorar e aplicar os princípios e práticas".

Este grupo publicou, em forma de manifesto (Manifesto Ágil, em português, 2001), doze princípios que julgavam importantes para que o desenvolvimento de software pudesse acontecer de forma mais rápida e com maior grau de iteratividade, interatividade e produtividade. Como as orientações propõem entregas mais rápidas de componentes, seria possível perceber erros ou equívocos ainda durante o processo a tempo de refazê-los e dar continuidade à escrita do código com mais segurança e rapidez

> Estamos descobrindo maneiras melhores de desenvolver software, fazendo-o nós mesmos e ajudando outros a fazerem o mesmo. Através deste trabalho, passamos a valorizar: Indivíduos e interações mais que processos e ferramentas. Software em funcionamento mais que documentação abrangente. Colaboração com o cliente mais que negociação de contratos. Responder a mudanças mais que seguir um plano, ou seja, mesmo havendo valor nos itens à direita, valorizamos mais os itens à esquerda (Manifesto Ágil, em português, 2001).

Nesse movimento já se fazia mais urgente se voltar a um olhar mais atento às situações interacionais e de comunicação que acontecem durante o desenvolvimento de software. Os participantes do manifesto incentivam esses intercâmbios visando preencher uma lacuna percebida por usuários de modelos mais tradicionais de DS para os casos onde o problema, equívoco conceitual ou erro é descoberto apenas ao final do processo de escrita do código e entrega ao cliente.

De acordo com Pressman (2011, p. 86), os que defendem metodologias ágeis reforçam que devem existir os chamados por ele de "traços-chave" entre os atores de um time ágil e do time entre si, que são: competência, foco comum, colaboração, habilidades de tomada de decisão e de solução de problemas confusos, auto-organização, confiança mútua e respeito. Pressman já vislumbrara, ali, a importância de imbricações de saberes e competências que iriam além das habilidades chamadas técnicas, conhecidas pelos clássicos, e até então suficientes para resolver questões globais de DS. Para buscar melhores soluções para estas questões mais complexas foi criado O SCRUM.

O SCRUM é um tipo de metodologia ágil muito utilizada no Brasil. Tem uma terminologia própria e consolidada na área e, de acordo com Oliveira, Guimarães e Fonseca (2013), possui aderência aos princípios do manifesto Ágil, podendo ser descrito a partir dos papéis de três atores principais,

> O SCRUM possui três principais papéis: *Product Owner*, *Scrum Master* e *Scrum Team*. O primeiro tem a responsabilidade de definir quais serão as diretrizes principais do projeto (*Release*) e requisitos de alto nível (*Product Backlog*), priorizar de acordo com as necessidades do mercado e aprovar ou reprovar as entregas feitas. É ele quem define o *Sprint* e *Release Goal.* O *ScrumMaster* é o líder do SCRUM e é responsável por garantir que o Scrum Team trabalhe em condições adequadas, removendo impedimentos, solucionando dúvidas e assegurando a produtividade de cada membro. Além disso, deve garantir que o processo definido está sendo seguido e sugerir à área de Gerência de Processos possíveis alterações no processo. Já o Scrum *Team*, é uma equipe auto-organizada responsável pelo produto gerado ao final de cada ciclo (*Sprint*), que estima o tamanho de cada requisito (*Selected Backlog*) a ser implementado e se compromete em atingir o *Sprint* e *Release Goal* definidos. (OLIVEIRA; GUIMARÃES; FONSECA, 2013).

Essa mudança técnica foi muito evidenciada dentro da literatura e questões, tais como sua confiabilidade para resolver os problemas levantados durante o DS, bem como sua eficácia e aderência às metodologias clássicas vem sendo discutidas entre pesquisadores (OLIVEIRA; GUIMARÃES; FONSECA, 2013; FERREIRA; LIMA 2005; HIRAMA, 2012). Porém, mesmo a Metodologia Ágil dando um passo nas questões de iteratividade e interatividade, ainda não contempla de forma interligada e clara a indissociabilidade entre os elementos técnicos e sociais no processo de desenvolvimento de softwares.

Neste sentido, percebemos que a trajetória clássica e técnica da ES e do DS rumo a encontrar processos que consigam minimizar as situações complexas

que se dão no desenvolvimento de softwares, evidencia que as opções por modelos rígidos e lineares de controle dos processos prevalecem e que, mesmo os modelos que se propõem a serem mais dinâmicos, também não conseguem se estabelecer de forma a resolver as queixas de clientes e Desenvolvedores para os problemas vivenciados por eles.

No próximo tópico aprofundaremos na proposta que nos parece a mais acertada para DS a partir do momento que busca dinamicidade e praticidade sociais e técnicas, de forma não fragmentada. Sem perder a essência do produto almejado e aproveitando, ao máximo, as competências e saberes acumulados ao longo do tempo nas diversas áreas do conhecimento, principalmente as sociais.

Essas propostas vieram de uma linha das Ciências Sociais e vem se destacando no Brasil a partir da proposta de um "Olhar sociotécnico para a Engenharia de Software", como uma forma de auxiliar Desenvolvedores, Engenheiros e demais atores do processo de DS na utilização de novas perspectivas de trabalho multidisciplinar (CUKIERMAN; TEIXEIRA; PRIKLANDNICKI, 2007).

2.4 Modelos industriais de melhoria de software e Modelos Sociais: desafios e conquistas

A Engenharia e Engenheiros de Software, mesmo em suas abordagens mais tradicionais de descrição de processos e modelos de DS, acabam por exprimir ou citar a importância das trocas e comunicações entre os envolvidos na ação de desenvolvimento. Porém, não conseguem demonstrar na prática, quais seriam os espaços específicos de diálogos e saberes que propiciariam ou ajudariam para tradução de uma prática social a qual se deseja transformar em um software.

Butler e Johnson (2015), ao desenvolver estudos em DS, afirma que transformar um projeto de software em um produto, engloba etapas importantes e que a etapa comunicacional (decisão por fazer o software, comunicação com o grupo sobre o produto almejado, início e entrega do software, dentre outros), envolve

prévias onde o código deve ser o componente mais acompanhado do processo, porém não é sua maior parte.

Estudos desenvolvidos nas áreas de Engenharia e DS (LEAL, 2008; FERREIRA; LIMA, 1996), chamam a atenção para esse fato alertando que os variados aspectos a serem considerados durante o planejamento de um software podem ter pontos frágeis e que modelos industriais de melhoria de software, podem não ser suficientes para seu sucesso (LEAL, 2008).

De acordo com esses estudos, metodologias sociotécnicas podem propiciar uma melhor aproximação do produto idealizado ao produto final, sendo importante que haja um esforço no sentido de "traduzir" as diferenças técnicas ou sociais relativas ao objetivo do projeto para que a colaborações ocorram, (ALBUQUERQUE; SIMON, 2009).

É seguindo esse entendimento que hoje a Engenharia de Software e também outras áreas do conhecimento, já percebem alguns efeitos colaterais do uso de metodologias que almejam cobrir um número maior de usuários em detrimento às características necessárias ao atendimento do usuário final. Também já perceberam que imbricações sociais, políticas e contextuais fazem parte de uma complexidade que permeia o modo como a sociedade e o mundo se constroem e se perpetuam e que esses elementos devem fazer parte da gênese do trabalho da ES e do DS (CUKIERMAN; TEIXEIRA; PRIKLANDNICKI, 2007).

2.4.1 Estudos sociotécnicos no Brasil

No Brasil, propostas de um mergulho mais profundo nessa temática como forma de abrir caminhos para uma formação que torne factível a percepção que a ES necessita trabalhar além dos muros da técnica, trouxeram para o campo acadêmico a junção das áreas exatas e sociais propondo uma forma mais contextualizada de se pensar a Engenharia de Software.

Trabalhando de forma estabelecida e desde 2005, a partir da criação do evento WOSES (Um Olhar sociotécnico sobre a Engenharia de Software), temos grupos de estudos com profissionais de várias áreas, que defendem uma postura sociotécnica para a ES brasileira. Esses grupos vêm se interessando em procedimentos de ES, que integrem em seus processos os aspectos técnicos e não técnicos do desenvolvimento de softwares em suas nuances sociais e humanas (TEIXEIRA, 2007; LEAL, 2008; CUKIERMAN; TEIXEIRA; PRIKLANDNICKI, 2007).

> O WOSES se propõe como um espaço pioneiro no Brasil dedicado a investigar as possibilidades e potencialidades de um olhar sociotécnico especificamente lançado sobre as práticas de desenvolvimento e uso de software, em sua busca de projetar e desenvolver software de alta qualidade. Um olhar que busca apreender o desenvolvimento de software sem fragmentá-lo em "fatores ou aspectos técnicos" de um lado, e "fatores ou aspectos não-técnicos" de outro, sem fatorá-lo em quaisquer outras dualidades ("fatores técnicos" versus "fatores humanos, organizacionais, éticos, políticos, sociais, etc.") que terminem por desfigurar o "pano sem costura" que imbrica no desenvolvimento e uso de software o técnico e o social em um mesmo e indivisível tecido. (Abertura WOSES 2009)

Nas palavras deste grupo, o esforço necessário para o desenvolvimento de sistemas apresenta problemas e desafios de complexidade muito além da técnica, ou seja, que exige a intervenção de saberes diferenciados, oriundos de outras áreas do conhecimento. Por essa razão, a comunidade de ES não pode deixar de se "contaminar" pelas contribuições das ciências humanas e sociais. Felizmente, muitos pesquisadores têm percebido que a tecnologia da informação também é inevitavelmente social e, com isso, têm tentado focalizar a ES como um problema concomitantemente de complexidade social.

Conforme Cukierman, Teixeira e Prikladnicki (2007),

> As novas tecnologias modificam a forma e a substância do controle, da participação e da coesão social. Porém, ao fazê-lo, são também modificadas pela experiência social. É o caso da ES (Engenharia de Software) e, de fato, sua comunidade acadêmica e profissional usualmente destaca a importância de questões sociais, culturais, políticas e organizacionais nos projetos de desenvolvimento de software e nos projetos de implantação e melhoria de processos de software.

O grupo também reconhece as dificuldades que permeiam imbricações, principalmente, entre áreas distintas do conhecimento, *"Certamente que a empreitada de fazer dialogar saberes tão distintos pode mostrar-se extremamente dificultosa, não sendo, todavia impossível, e, portanto, é recomendável que seja experimentada"* (CUKIERMAN; TEIXEIRA; PRIKLADNICKI, 2007, p. 206).

Livre da pretensão de fazerem comparações entre as experiências estrangeiras e brasileiras, a proposta é de um aprofundamento e estudo sistemático do tema, onde fiquem cada vez mais claras e produtivas algumas proposições de se avançar nas discussões e relativas a uma experiência sociotécnica brasileira.

Assim, estudos apresentados por pesquisadores da linha de tecnologia e sociedade vem evidenciando que a indissociabilidade entre elementos técnicos e não técnicos deve ser advogada pelos atores dos processos que envolvem ES, e ressaltam que tal prática, ao se consolidar, pode trazer melhorias no cotidiano das empresas diminuindo situações e impasses que ocorrem quando se tenta trabalhar de forma fragmentada alienando esses elementos.

Por fim, entendemos como muito importante para a ES se apropriar desses conhecimentos bem como também temos a oportunidade, enquanto país, em nos destacar pela pesquisa em diversas áreas e sermos protagonistas de estudos que reverberem um saber que avança para a melhoria do Desenvolvimento de Software como um todo.

2.5 Mundos sociais distintos no DS: a um passo para a compreensão de uma experiência sociotécnica

Conforme indicamos na introdução, a ambição desta pesquisa é a de compreender alguns entendimentos, tensões e conflitos advindos da experiência em se implantar o modelo MPS.Br e desenvolver um software novo a partir do entendimento de como bases sociotécnicas e os estudos sobre formas de trabalho em conjunto e comunidades de práticas poderiam cooperar para um melhor resultado da proposta empreendida.

Para tanto, fez-se necessário apreender como se deu a corrente mundial que culminou na criação de processos conhecidos como tradicionais ou clássicos para o DS e também como as ciências sociais vêm contribuindo para uma nova forma de pensamento e aplicação pela ES de formas que contemplem outros saberes no momento da concepção e Desenvolvimento de Softwares.

Essa busca nos despertou para a importância dos estudos de Barrett e Oborn, que publicaram, em 2010, um material significativo no qual analisaram o comportamento em equipes de software lotadas em espaços geográficos distantes e oriundas de diferentes culturas: Desenvolvedores que estavam atuando em uma empresa sediada na Jamaica e Desenvolvedores que atuavam em uma empresa localizada na Índia.

A partir do acompanhamento dos trabalhos dessas equipes, Barrett e Oborn, (2010) examinaram o uso de requisitos de software e ferramentas de gerenciamento de projeto como pontos convergentes de comunicação utilizados pelos Desenvolvedores para facilitar a partilha de conhecimento entre os envolvidos no projeto de desenvolvimento de um software.

Os autores avançaram na constatação que, mesmo em ambientes complexos, como o caso de DS feito por grupos culturalmente diferentes os objetos de convergência para a comunicação poderiam tanto facilitar a colaboração entre times profissionais quanto promover situações de conflito entre eles. Em suas pesquisas sobre o DS feito por equipes interculturais, Barrett e Oborn, (2010), verificaram que a utilização de requisitos de software na construção conjunta do conhecimento funcionou para esse grupo até certo momento.

Estes verificaram ao final do estudo, que as diferenças culturais coligadas aos conflitos interculturais e aliados à necessidade de poder e controle comum à maioria dos atores humanos, acabaram por inibir a troca de conhecimentos entre o "*team"* e, consequentemente, obstaram o trabalho colaborativo conjunto desejado ao processo. Mesmo assim, os autores acreditam que o tema deva ser

melhor estudado para que a experiência não seja generalizada para qualquer tipo de relações interculturais.

Thomas, R; Sargent L; Hardy (2008) endossam essa constatação ao destacarem que as partes interessadas e os grupos sociais têm diferentes acessos aos recursos, autoridade e de definição no que se refere aos pontos de comunicação dos grupos. Eles reforçam tais conceitos ao afirmarem que é a moderação entre tensões e ambiguidades relativas a esses objetos que permitem a justaposição satisfatória para tornar a informação reconhecível pelos atores em seus diferentes grupos. Porém, esta justaposição estaria no limite suficiente para flexibilizar as interpretações advindas de diferentes mundos sociais.

Nesta perspectiva os autores destacam que as negociações e trocas de conhecimentos acionados por um ou mais objetos de comunicação pode mudar, embora o próprio objeto possa permanecer inalterado. Destacam ainda que estes pontos podem ser observados a partir de uma visão mais pluralista na qual as dinâmicas políticas, as tensões e os conflitos acontecidos a partir da sua produção também interferem em seu potencial colaborativo.

Um ponto interessante dessa discussão é o entendimento que objetos, enquanto artefatos ou conceitos operacionalizáveis, fazem parte de um sistema dinâmico e interacional, podendo ser renegociados e/ou contestados durante o desenvolvimento dos processos organizacionais.

Enfatizamos isso porque, ainda no *case* em Barrett e Oborn (2010), o olhar atento do gerente de equipes permite observações a serem feitas ao longo do processo as quais podem ajudar no entendimento de "como diferentes elementos interacionais, tais como aumentar os mecanismos de autoridade e controle gerenciais, afetam os pontos de convergência comunicacionais e o desenvolvimento da equipe".

Para defender esses pontos de convergência, Barrett e Oborn (2010), ressaltam que, no caso da pesquisa citada, as tensões e controvérsias geradas ao longo do processo,

> Comprometeram o desempenho das equipes evidenciado pela baixa eficiência dos dois grupos, pelos atrasos significativos no projeto, pela má qualidade (demonstrado pela reescrita de código), e pela inviabilização da equipe com os grupos entrincheirados nas posições de "nós" e "eles" durante a execução do projeto (BARRET; OBORN, 2010).

Com esses elementos e conhecimentos em mãos, nosso trabalho busca visualizar, no processo estudado, momentos em que o pensamento não fragmentado se configurou em situações de tensões, de sucesso e satisfação dos grupos e comunidades como também, estender essa linha ao tratar do trabalho em conjunto, comunidades de prática, alienação e divisão do trabalho e objetos mediadores do trabalho em conjunto, nosso pano teórico o qual desenvolveremos no próximo capítulo.

3 ABORDAGEM DE PESQUISA

Esse capítulo é dedicado à apresentação dos estudos dos diferentes tipos de trabalho em conjunto que podem ocorrer durante as dinâmicas das questões fundantes dessa pesquisa. Essas abordagens e seus conceitos serão cruciais para o entendimento dos estudos de caso e contribuem para ressaltar a validade da aplicação desses conceitos para este trabalho. Traremos aqui alguns tópicos sobre Comunidades de Prática, Limites e Formas de Trabalho Coletivo e seus desdobramentos para melhor entendimento das falhas que ainda ocorrem durante os processos de DS.

3.1 Comunidades de Prática e mundos sociais distintos

Comunidade de prática é uma proposta de aprendizagem conjunta criada e divulgada por Etienne Wenger e Jean Lave, em 1991. Uma comunidade de Prática acontece, de acordo com os autores, quando,

> Um grupo de pessoas informalmente unidas pelas experiências compartilhadas e pela paixão por empreendimentos conjuntos se juntam de forma organizada compartilhando suas paixões, seus interesses, um conjunto de problemas ou um entusiasmo por um assunto específico de forma a aprofundar o seu conhecimento e expertise na área por interações regulares.

Para dar corpo a essas proposições, Wenger (2006) esclarece que comunidades de prática estão inseridas nas teorias sociais de aprendizagem. De acordo com ele, a aprendizagem não pode acontecer apenas pelo acesso aos termos, palavras ou orações que definem uma área e que deve permitir ao indivíduo envolver-se na experiência do conhecimento para que seja significativa e não simplesmente traduzi-lo a orações ou palavras. Wenger (2006), faz um amplo apanhado das questões que permeiam as situações de aprendizagem entre grupos com interesses similares, desde o advento da Revolução Industrial até a atualidade. De acordo com ele, o pressuposto estabelecido é que a aprendizagem pode ser verificada quando existe uma afinidade especial entre o "perito e o novato no interesse comum sobre o mesmo tema ou situação em particular." Neste sentido,

Wenger (idem), declara que se trata de uma Teoria Social de Aprendizagem e não uma Teoria de Aprendizagem Social, uma vez que ele compreende que os indivíduos são "seres sociais envoltos em práticas sociais".

Essa teoria ajuíza uma aprendizagem que busca e valoriza as experiências por traz das frases ou palavras que dizem de um tema e prevê um processo de participação social enquanto prática (como o desenho de um conjunto de relações entre o novato e os outros membros do grupo, a prática do grupo e a cultura do grupo). Wenger (idem) discorre ainda sobre seu conceito de aprendizagem significativa, a qual deve permitir ao indivíduo envolver-se na experiência de conhecimento. Wenger, (2006), entende que, a partir do conhecimento e do uso de ferramentas de construções de significados vindos das práticas sociais específicas de um grupo ou comunidade é possível identificar grupos que podem se enquadrar no conceito de Comunidade de Prática, bem como se reconhecer como tal. (FIG. 4).

Em relação às organizações e suas relações de trabalho, Wenger (idem) considera que toda organização possui uma miríade de comunidades de prática inseridas nela e que por esse motivo devem estar atentas às formas como as relações e as construções do conhecimento se dão nesses ambientes. Ele defende a importância das pessoas que realizam atividades comuns nesses contextos bem como a capacidade desses grupos em se organizarem para interferir de forma positiva no aprendizado do grupo e também nas formas com as comunidades aprendem. Como resultado ele apresenta um cenário com ferramentas para compreender a aprendizagem como um processo participativo e de natureza interativa e compartilhada.

FIGURA 4 - QUADRO COMPARATIVO COMUNIDADES DE PRÁTICA

	Qual é o Objetivo?	Quem Participa?	O que têm em comum?	Quanto tempo duram?
Comunidade de Prática	Desenvolver as competências dos participantes; gerar e trocar conhecimentos	Participantes auto-selecionados	Paixão, compromisso e identificação com os conhecimentos especializados do grupo	Enquanto houver interesse em manter o grupo
Grupo de Trabalho Formal	Desenvolver um produto ou prestar um serviço	Qualquer um que se apresente ao gerente do grupo	Requisitos do trabalho e metas comuns	Até a próxima reorganização
Equipe de Projeto	Realizar determinada tarefa	Empregados escolhidos por gerentes seniores	As metas e pontos importantes do projeto	Até o final do projeto
Rede Informal	Colher e transmitir informações empresariais.	Amigos e conhecidos do meio empresarial	Necessidades mútuas	Enquanto as pessoas tiverem um motivo para manterem contato.

Fonte: Antonello; Ruas (2005).

Os estudos de Etienne Wenger são uma contribuição contemporânea para um avanço no entendimento das novas configurações de aprendizagem entre grupos e as oportunidades de crescimento, aprendizagem e sobrevivência organizacional. A FIG. 4 ilustra as diferenças entre as formas como grupos se reúnem na atualidade e a partir de seus objetivos. A Comunidade de Prática se destaca pela geração de conhecimentos que dialogam com o interesse comum de pessoas que se interessam por um tema e de como elas se organizam para trocar informações e aprendizagens.

A área de Desenvolvimento de software, por ser muito dinâmica possui muitos desenvolvedores envolvidos por uma comunidade de prática ou mais. Na próxima sessão estudaremos o tema Formas de Trabalho em Conjunto, no qual destacaremos a colaborativa, que acaba por facilitar a formação de Comunidades de Prática como opção de construção do conhecimento.

3.2 Formas de Trabalho em Conjunto

Em sua tese objetos de fronteira, limites e formas de trabalho conjunto, Rajão (2011), nos oferece uma descrição importante sobre tipologias de trabalho

conjunto, e a importância em se diferenciá-los para uma compreensão da influência de cada um durante a execução de uma tarefa ou projeto.

Desses tipos citados e também descritos em modelos organizacionais, encontramos: coerção, coordenação, cooperação e colaboração (RAJÃO, 2011; ENGESTROM, 1997; POWEL, 1990; GRAY, 1989).

A coerção é o tipo descrito nas pesquisas e indica uma relação de trabalho em que o indivíduo exerce pouco ou nenhum espaço de decisões e trocas, com um espaço comunicacional extremamente reduzido, focado apenas na realização, geralmente focal, das tarefas a serem realizadas pelo grupo.

Já mais conhecida e executada como forma de trabalho em conjunto, a coordenação, vai além da simples execução de ordens segundo Collins (2007 apud Rajão 2011). Geralmente ocorre dentro de uma linha hierárquica formal para Powell, (1990), sendo um desenho organizacional muito comum nas instituições. Consiste na delegação ao grupo do trabalho a ser realizado.

Neste modelo, o coordenador acompanha e controla as atividades mantendo uma relação hierárquica vertical que pode ser feita de forma presencial ou à distância. Essa modalidade de trabalho em conjunto, tangencia-se com o tipo coercitivo de trabalho, uma vez que, nem sempre, impõe a criação de formatos mais densos de compreensão mútua. Para esses casos, as situações em que ocorre a interação entre coordenador e coordenados se limitam, frequentemente, a especificações de trabalho, relatórios e indicadores de trabalho (ADLER; HECKSCHER, 2006).

Novamente, esse tipo de relação de trabalho se assemelha ao tipo coerção no momento em que os participantes das equipes envolvem-se pouco com o trabalho dos colegas permanecendo concentrados em desenvolver suas próprias tarefas e especificações de trabalho. Atraente pelo fato de os grupos ou equipes serem livres para manter certa independência enquanto executam suas tarefas, a cooperação prevê grupos mais abertos e não ligados por relações hierárquicas formais ou diretas.

Para contribuir com essa linha de estudo, Engestrom, et al (1997), colocam que isso não evita que tais grupos se expandam em buscar práticas compartilhadas com outros grupos ou práticas de fronteira. Porém, esses grupos ainda estão focados no cumprimento e entrega de sua tarefa focal. São parceiros, mas não necessariamente têm por postura a criação de práticas conjuntas.

A colaboração, por outro lado, evidencia uma forma de trabalho que vai além do cumprimento de ordens, tarefas, especificações e roteiros modelares. Essa modalidade, de acordo com Brown e Duguid (1991), provoca as formas tradicionais de trabalho (coerção, coordenação, cooperação), em seu desafio mais preciso: as transformações que podem advir de movimentos comprometidos com a comunicação entre grupos de forma mais pontual e ativa.

Engestrom (2001 apud Rajão, 2011), reforça que as formas de trabalho em conjunto mais utilizadas pelas organizações na atualidade "não são suficientes para trazer inovação, lidar com profundas contradições" e que, "ao invés de uma proposição idealista, a colaboração tornou-se um imperativo em organizações complexas" (ADLER; HECKSCHER, 2006).

Nesse argumento, ele demonstra que, para lidar com organizações de cunho cada vez mais complexo as peculiaridades que vão diferenciar, principalmente, o tipo de trabalho de "cooperação" do tipo "colaboração", são as mais importantes de se ter claras para compreender as organizações da contemporaneidade.

Para o pesquisador essa visão é imprescindível, uma vez que o estudo das comunicações entre os Desenvolvedores e seu insumo para trabalho, os requisitos, estão intimamente ligados a processos de gestão, hierarquias, jogos de poder, autonomia das equipes e outros e que uma mesma empresa vivencia essas tipologias concomitantemente. O que nos levar a aprofundar um pouco nos estudos de Karl Marx, sobre alienação do trabalho.

3.3 Karl Marx: Alienação e divisão do trabalho

Karl Marx possui um trabalho profundo e extenso acerca do trabalho, considerando sua divisão e também as relações que o indivíduo possui entre si quando está perante a situações de divisão do trabalho. Marx estudou essas relações traçando paralelos para elas desde a propriedade como existia em povos primitivos até o momento da eclosão da revolução industrial e das novas formas em que a produção era pensada e praticada, principalmente no que se refere ao produto do trabalho, que agora não pertencia mais ao artesão e sim ao dono da fábrica (GARAUDY, 1967).

Esse salto radical de como as formas de produzir se transformaram podem ser ilustradas também segundo Marx (1964), por um momento em especial: quando o artesão, que detinha as fases de reflexão e execução do seu trabalho passou a fazer parte de processos fragmentados de produção, distanciando-o, de certa maneira, da sensação de realização próprias ao resultado da execução de um trabalho planejado e efetuado por um mesmo indivíduo. Ficou então, para o trabalho industrial, dentro de seu espelho capitalista de produção, consolidar essa divisão entre trabalho manual e o intelectual.

Para dar forma à sua lógica de pensamento, Marx (1964) introduz o conceito de Alienação do trabalho, aonde, num primeiro momento pré-capitalista vem, sem que haja, ainda, uma ruptura total entre o fazer do trabalhador e seu saber. Porém, o mesmo não acontece quando a forma industrial de trabalho começa a imperar e os processos passam a ser fragmentados ou automatizados. Nessa forma de produção, o produto final é resultado de um "rejunte" e o trabalhador é parte do capital de produção e de seus objetivos.

Assim, para a situação de alienação total, quando se percebe a quebra entre as reflexões necessárias a empreender uma tarefa e sua feitura em si, Marx (1964), sinaliza que existe, ali, uma apropriação do capital. É interessante ressaltar que essa lógica acaba por instituir uma configuração de trabalho de relações hierárquicas que modularão as futuras relações entre os trabalhadores que agora

serão divididos entre os técnicos científicos, que vão organizar e acompanhar os processos de trabalho e os operários ou executores da tarefa, que utilizarão do processo ou modelo para executar a tarefa planejada.

Como conclusão Marx (1964) salienta que a consequência percebida ao se organizar um modo de produção que possui o capital como único mote para a sua consumação, é uma apropriação cada vez maior do saber social (antes em poder do artesão ou trabalhador manual) de modo a aumentar os distanciamentos entre a reflexão do que se faz e sua feitura. De acordo com o autor, a consequência disso seria a sensação por parte do trabalhador de não mais fazer parte do resultado daquele processo de produção (alienação).

Estas considerações nos ajudarão e entender melhor fenômenos da produção industrial da atualidade e novos modos de organização entre os indivíduos de forma a criar alternativas para preservação do saber global do processo, como veremos abaixo.

3.4 Objetos mediadores do trabalho conjunto, razão instrumental e razão comunicativa

As formas de trabalho em conjunto são elementos importantes quando consideramos estudo e análise de arranjos organizacionais, desde os mais simples aos mais complexos. Isto porque elas envolvem engajamentos sobre como a organização ou grupo se estabelecem para representar seus papéis sociais e profissionais dentro de escopos e limites pré-estabelecidos. Neste sentido, é importante conhecermos um pouco mais sobre alguns objetos considerados mediadores do trabalho em conjunto, uma vez que essas linhas são o pano de fundo teórico para melhor compreensão de nosso percurso empírico bem como seus resultados. Para dar reforço a essa proposição, visitamos o trabalho de Jurgen Habermas e Max Horkheime que instituíram os conceitos de Razão Comunicativa, (HABERMAS, 1987), em contraponto à Razão Instrumental (HORKHEIME apud HABERMAS, 1987).

Sobre a razão instrumental, podemos dizer que ela acontece quando se entende o conhecimento como uma ferramenta de poder, capaz de controlar e/ou dominar a natureza e outros seres humanos (HORKHEIME apud HABERMAS, 1987). Nessa lógica de pensamento, o saber científico sai de seu status de ser provedor de acesso social a conhecimentos "verdadeiros", e passa a ser uma ferramenta que foge aos princípios iluministas da razão ocidental. A razão instrumental busca, por meio dos conhecimentos gerados pela ciência, via diversos mecanismos, formas de subjugação e exploração do outro a partir do valor operacional incutido a ela.

Já a razão comunicativa é uma contribuição de Jurgen Habermas, filósofo e sociólogo alemão, que vai de encontro à teoria da razão instrumental de uma forma mais positiva, no sentido que indica que a informação e o conhecimento da forma como são gerados socialmente nem sempre têm a intenção de seguir uma proposição coercitiva e nem sempre servem para atender a pré intenções de dominação ou poder. Habermas contribui muito ao inserir o conceito de razão comunicativa enquanto opção conceitual para formas livres, racionais e críticas de lidar com o conhecimento e informações. Essa maneira de se pensar a racionalidade abre espaços para estudos da comunicação que podem elucidar questões em que o desejo de se comunicar e trocar informações saem do padrão instrumental para algo mais amplo, colaborativo e sistêmico.

As abordagens teóricas apresentadas acima fornecem elementos importantes que poderão ser úteis para a compreensão do estudo de caso sobre desenvolvimento de software e introdução do MPS.Br na empresa. Em particular ao definir a Comunidade de Pratica e enfatizar o modo hermético que os membros dessas comunidades aprendem e transmitem informação, é possível compreender melhor as dificuldades de se relacionar entre as comunidades, entre gestores, DS e especialistas de domínio.

Por isso, além dessa definição foi útil também apresentar os desafios com relação ao Trabalho Conjunto onde, nessa literatura, demonstra-se como diferentes formas de trabalho em conjunto seja ela mais voltada para a colaboração onde há de fato uma troca de conhecimento indo até a coerção,

obtém-se resultados diferentes. Ligado a isso temos os estudos que descrevem como podemos ter formas de colaboração que são baseadas em ações comunicativas e que buscam a aprendizagem mútua indo na direção de formas mais coercivas que levam à alienação como definida por Marx e formas instrumentais de comunicação.

Abaixo traremos detalhes do nosso trabalho empírico e observações ao longo da pesquisa, com o olhar voltado para como se deu o processo de implantação de um modelo de melhoria de produto de software, o MPS.Br, que chamaremos de modelo clássico, juntamente com a construção de um software novo cuja produção deveria seguir esses padrões.

4 DAS ORIGENS AO SONHO DA FÁBRICA DE SOFTWARE

Desde a década de 60 até o momento atual, que o termo *Fábrica de Software* ocupa de alguma forma os estudos, aplicações ou desejos de equipes de DS. Isso se dá porque, mesmo com o passar dos anos, a melhor forma de se produzir um Software, ainda está ligado à implantação e domínio de certas regras construídas para esse fim.

Para contar como nossa pesquisa aconteceu, utilizaremos o acompanhamento do planejamento e resultados de partes das entregas do setor de DS para a construção de um software de Bibliotecas. Essa demanda veio de uma agência de desenvolvimento de softwares de gerenciamento sediada em Belo Horizonte.

O software estudado é um software de bibliotecas que está sendo construído a partir da decisão de se aplicar um modelo clássico, o MPS.Br, como guia para qualidade dos produtos futuros da empresa. O modelo foi sugerido pelo setor de DS após um pedido da gestão da empresa por um maior controle das documentações dos softwares criados e, também, de um controle da rastreabilidade dos códigos desenvolvidos.

Mesmo sendo um desejo manifestado por todo o grupo de DS, muito foi pensado e discutido pelos gestores e sócios da organização para se chegar à decisão pelo uso do modelo MPS.Br. A empresa funcionava de forma mais aberta e interativa antes da aplicação do modelo, mas, o crescimento da aceitação pelos seus produtos e a demanda pela escrita de outros fomentaram a necessidade de se mudar a forma como softwares eram desenvolvidos por ela.

Esse cenário acabou por criar entrelaçamentos que fizeram o pano de fundo para nosso trabalho onde, de acordo com os atores entrevistados, trouxeram impedimentos para que as expectativas de produtividade e qualidade com a implantação do modelo não ocorressem.

No próximo capítulo, conheceremos melhor a empresa estudada, seu posicionamento no mercado mineiro e a forma que a empresa trabalhava antes da proposta de implantação, certificação e utilização de um modelo formal de qualidade do produto, o MPS.Br e suas controvérsias. Apreendemos para o caso da empresa pesquisada, que esses conflitos ocorrem desde o primeiro planejamento com requisitos em formato de princípios institucionais até os de ordem direta e acabam por distinguir o produto final do produto idealizado. Esses trajetos e suas implicações serão nosso foco.

4.1 Primeiros produtos e criação da "marca"

A empresa pesquisada é uma organização de pequeno porte, sediada na cidade de Belo Horizonte, Minas Gerais e atua há mais de 16 anos no mercado local e nacional,

> "Aonde vem oferecendo aos seus clientes soluções em serviços de desenvolvimento e implantação de softwares gerenciais, realizando projetos de gestão e controle e criando ferramentas e soluções concretas e objetivas para atender às demandas do mercado" (Gestora).

De acordo com o WEB site da empresa, eles têm como missão "Promover as melhores soluções tecnológicas mantendo a confiança conquistada junto aos clientes e apoiando o crescimento e desenvolvimento estratégicos das empresas parceiras, acompanhando a evolução tecnológica, prestando serviço de excelência e seguindo as necessidades do mercado".

Formada inicialmente por dois Analistas de Sistemas e um Engenheiro, a instituição foi fundada em 1999 e iniciou suas atividades com o objetivo de levar aos seus clientes soluções personalizadas através de softwares feitos sob demanda.

No ano de 2001 a empresa amplia e engloba em seu corpo societário um profissional da área de Ciência da Informação, objetivando atender uma necessidade dos novos clientes (empresas que dão manutenção predial à Caixa

Econômica Federal). O projeto era grande e foi feito em parceria com colaboradores do banco. Nasce o sistema BancoTeka (Nome fictício).

Esse novo sócio também veio para acompanhar a escrita de um produto de gerenciamento de serviços para a área educacional. Isto porque, a mensuração em massa das proficiências do alunado brasileiro estava expondo negativamente os níveis educacionais dos estudantes e as escolas, buscavam atender seus clientes com tecnologias educacionais mais atraentes.

É nesse contexto mercadológico educacional que nasce o primeiro software de gerenciamento de Bibliotecas da empresa, o SofTeka (Nome fictício). A empresa segue e, em 2002 dois novos sócios entram para o quadro, um na área de administração, outro na área de tecnologia. O corpo societário da empresa também faz parte do corpo funcional e essa configuração impactou nas formas como a empresa atuava até então.

Nesse percurso, os produtos SofTeka e BancoTeka são concluídos e entram em evolução para melhorias. Esses softwares passaram para o quadro principal de vendas e contribuíram para uma guinada estratégica da empresa que consistiu em ressignificar sua marca e seus futuros produtos.

Em 2009 e contando com 10 anos de experiência no mercado de gerenciamento é criada a marca *Yes* (Nome fictício). Um novo sistema é desenvolvido, o YesManutenção cuja construção contou com a participação de engenheiros especializados e de renome nacional na sua concepção. Nesse mesmo ano, a empresa agrega prestadores de serviços de grandes instituições como o Banco do Brasil, Itaú Unibanco e FIAT aos prestadores de serviço da Caixa Econômica Federal.

Ainda em 2009, a organização inicia suas atividades na área da responsabilidade social, com o apoio à criação da Associação de Gestão e Engenharia de Sistemas de Informação (AGESIN). Hoje qualificada como uma OSCIP nos âmbitos Estadual e Federal. A criação da AGESIN foi o resultado da união de profissionais de diversas áreas do conhecimento com *"a finalidade de estimular*

a prática da cidadania e a busca coletiva de soluções para desafios comuns, visando à construção de uma sociedade voltada para o crescimento sustentável e sólido" (Disponível em: <agesin.org.br>).

Entre 2010 e 2012 dois sócios da empresa se desligaram da sociedade vendendo suas cotas para o mercado. Entram duas novas sócias, com formação em gestão financeira e gestão de terceiro setor. Nesse ano, o novo quadro societário da Empresa compreendia cinco sócios nas áreas de Tecnologia, Gestão Financeira, Gestão Administrativa, Gestão do Terceiro Setor e Políticas Públicas e Ciência da Informação. Em 2012 foi consolidado outro novo software educacional, o YesPNLD (Nome fictício), com o objetivo de atender políticas públicas de distribuição de livros didáticos no país.

A empresa percebe que existe um nicho de mercado carente de produtos específicos das áreas de que tem expertise e resolve ampliar novamente seus negócios. O contexto econômico do país estava favorável e várias linhas de crédito eram oferecidas ao micro e pequenos empresários com juros e prazos de pagamento atrativos. A empresa possuía softwares estáveis e clientes fidelizados e acreditava em sua capacidade de escrita de produtos inovadores e de grande aceitação no mercado. Era hora de crescer.

Assim, em janeiro de 2013 o grupo societário decide mudar de sede, (de uma sala de 80m para um andar de 340m), e aumentar o numero de funcionários dos setores comercial e de DS. Tudo isso para dar espaço à atualização e aplicação de novas práticas e tendências de mercado. Nesse ano, foi decidido investir e empreender no desenvolvimento e lançamento de um novo produto, *o* YesTeka (nome fictício) o qual observaremos nesse estudo de caso.

4.2 Um sistema de bibliotecas inovador

"Integrar as informações constantes de um sistema tradicional de bibliotecas a uma Rede Social para Leitores, os quais poderão interagir entre si em ambiente WEB", (Web Site da empresa). Esse foi o desejo motivador para a modelagem de um software de gerenciamento de bibliotecas com funcionalidades diferentes das oferecidas no mercado tradicional da biblioteconomia. O objetivo para a inovação era o de interferir nas formas como as informações são disponibilizadas a estudantes e comunidades à época.

Para atender a esse propósito, o software precisaria ser desenvolvido a partir da concepção de um produto "*diferenciado no mercado de softwares de bibliotecas ao ligar as proposições administrativas, biblioteconômicas e pedagógicas do software a uma metodologia de interação social inovadora e contemporânea"* (Especialista de Domínio).

O objetivo era que o desenvolvimento do produto fosse capaz de unir os instrumentos e suportes necessários para acolher os investimentos financeiros destinados para a aquisição e controle de acervos físicos e virtuais de escolas, comunidades, acervos públicos e de organizações.

O software em questão seria uma *"Rede de Catalogação Cooperativa com dados tratados biblioteconomicamente acoplada a uma Rede Social de Leitores"*. Esse sistema deveria seguir padrões atualizados, nacionais e internacionais de interoperabilidade e Biblioteconomia. Esse sistema deveria conter, também, as prerrogativas comunicacionais apresentadas pelas redes sociais já consolidadas socialmente, tais como o Facebook, Moodle, sites de compras, sites de locação de filmes entre outros.

Além da inclusão de padrões já consolidados na área de Biblioteconomia, a empresa se propôs a fazer a junção das já conhecidas facilidades administrativas dos sistemas de gerenciamento de acervos, às funções tecnológicas, culturais e pedagógicas necessárias a essa nova proposta de construção de um sistema com

diferenciais. Tudo isso em um ambiente virtual "que permitisse trocas entre seus interagentes, devido ao seu perfil de imigrantes e nativos tecnológicos", Veras (2011). Para alcançar esse objetivo, ao falar sobre softwares de bibliotecas, considerou-se que:

> Uma das condições para que o desenvolvimento de um novo software de bibliotecas se transformasse em um projeto vendável, era a reflexão de como os recursos investidos em bibliotecas são efetivamente utilizados e aproveitados por bibliotecários, professores, educadores, estudantes e comunidades como apoio à promoção da proficiência, excelência leitora e letramento. (E-mail enviado à equipe de DS).

Essa era a mudança de paradigma esperada e, para tanto, a empresa buscou a ajuda de consultorias e de outras instituições que percebessem englobassem, a essa proposta, a participação não somente dos usuários administrativos do sistema, mas, também, a dos usuários finais, chamados no projeto de *interagentes* (os estudantes ou comunidades que seriam afetados pelas novas funcionalidades do sistema). Alguns estudos feitos por esses grupos demonstraram que:

> "houve um salto de acesso tecnológico muito grande nos últimos anos (tablets, celulares, aplicativos, nuvem, etc.) e que este salto afetou diretamente a forma como se aprende e como ocorre a mediação da informação nas escolas, em sua maioria ainda de cunho tradicional, incluindo-se, aí, as Bibliotecas". (Especialista de Domínio).

Por isso, era necessário delinear soluções para alcançar o público em geral e os nativos tecnológicos na forma como buscam, selecionam e usam a informação no mundo contemporâneo.

Foi a partir deste momento, "*levando-se em consideração tratar-se de um público dinâmico, com características co-criadoras e ao mesmo tempo migratórias"* que a equipe gestora da empresa considerou a descontinuação do software de bibliotecas já existente, o SofTeka, e a criação de um novo produto, o YesTeka, já em um modelo de documentação e rastreabilidade que se fizesse à luz de paradigmas de produção de sistemas de informação mais contemporâneos.

4.3 “Estórias” e Histórias: detalhes do desenvolvimento do software

Muito se falava sobre modelos e metodologias eficazes de DS. Construir softwares personalizados, com qualidade e de fácil interação era o carro chefe que elevou a posição da empresa como negócio no mercado. Seus softwares exigiam pouco uso do suporte técnico e mudanças e inclusões em seu escopo eram facilmente implementados.

Assim, em março de 2013 decidiu-se desenvolver um novo software para substituir o sistema para bibliotecas desenvolvido e comercializado pela empresa há mais de 10 anos. O SofTeka (software a ser descontinuado) é um sistema gerencial Desktop com módulo WEB que funciona para reservas, consultas e renovações de materiais bibliográficos. Seu público alvo são bibliotecas escolares, públicas, e comunitárias de médio porte. O sistema possui cerca de 200 clientes divididos em todo território brasileiro.

O YesTeka (software a ser desenvolvido) deverá ser totalmente WEB atender a protocolos de segurança da informação, usar o novo modelo de catalogação ainda não adotado em softwares concorrentes e possuir características de interatividade baseadas em tecnologias e conceitos que abarquem o uso de redes sociais e ambientes virtuais de aprendizagem pelos seus usuários, principalmente o público jovem (de 10 a 25 anos).

4.4 Tecnologias e sociedade: o percurso

As tecnologias que atendem jovens são muito dinâmicas. No que se refere a bibliotecas o país, até então, utilizava de formas básicas de controle de empréstimos e devoluções de livros aos usuários com alguns relatórios gerenciais e técnicos. Porém, o mercado sinalizava para um universo integrado e interativo para produção de tecnologias sociais.

Foi planejado um período de 18 meses para a construção, nominada pela gestão de "produto vendável". Para a entrega de seus módulos complementares, 24 meses. E, para viabilizar o tempo de execução do projeto, foram contratados dois Desenvolvedores experientes. Nesse cenário, e ainda na sua forma de trabalho original, algumas premissas começaram a ser definidas o YesTeka: Gestores, Engenheiro de Software, Desenvolvedores e Especialistas de Domínio foram informados dos prazos e reuniram para criar os eixos de escrita do futuro software.

Internamente tais eixos foram apelidados de os "10 mandamentos":

1. Era necessário que traçar um plano de ação que contemplasse Tempo versus Recurso financeiro versus o que se construiria;
2. Era necessário que existisse um completo registro das tarefas;
3. Um sistema Criativo, bonito, intuitivo, que encantasse e surpreendesse e trouxesse novidade ao mercado;
4. Um sistema que tivesse menos cliques para a execução das rotinas;
5. Um sistema que fosse fácil de fazer implementações e consertos;
6. Um sistema que não fosse engessado, duro, difícil de mudar algo;
7. Um sistema que desse ao usuário final e ao desenvolvedor a possibilidade de reutilizar informações e códigos;
8. Um sistema que tivesse a cara da equipe e da empresa e não só a vontade de bibliotecários ou de outro que tivesse mais desenvoltura em expor suas ideias;
9. Era fundamental saber que as metas seriam cumpridas;
10. Era fundamental saber que os prazos seriam cumpridos.

Existiam, porém, problemas a serem resolvidos. Necessário se fazia achar respostas para algumas questões: Como controlar as tarefas (demandas) a serem desenvolvidas? Como controlar os prazos (as entregas) destas demandas? Como documentar de forma mais completa? Como atender aos 10 eixos? Quanto custaria esse empreendimento?

Neste período foram criados e enviados a todos em letras maiúsculas mais três requisitos básicos que deveriam ser seguidos por todos os Desenvolvedores:

a) NÃO SE PODE CONSTRUIR NADA SEM ANTES CONSULTAR O SOFTEKA.
b) EM BIBLIOTECA TUDO SE ORDENA ALFABETICAMENTE OU NUMERICAMENTE.
c) TUDO DEVE SER FEITO LEMBRANDO QUE OS CLIENTES ATUAIS SERÃO MIGRADOS PARA O NOVO SISTEMA.

Era sabido por todos que existiam mais de 200 clientes que teriam que ser migrados. A gestora da empresa ponderou que o sistema antigo (apesar dos resultados da entrevista) era um bom sistema gerencial e o conhecimento e consulta a ele poderiam facilitar a compreensão do negócio. *"Então ficou para os desenvolvedores checarem sempre as regras do sistema SofTeka e usá-lo como espelho e também pensar o novo, melhorar, inovar, e mudar a imagem"* (Especialista de Domínio).

O projeto iria começar pela construção do módulo de Gestão da Biblioteca e a Rede Social ficaria para um segundo momento. Para consolidar essa proposta, a especialista de domínio, juntamente com a gestora da empresa, fizeram reuniões com os Desenvolvedores e apresentaram o sistema de gerenciamento de bibliotecas em desktop. Essas apresentações duraram dois dias, e foram mostrados alguns módulos do sistema aos novos Desenvolvedores e aos colaboradores do setor comercial, pois os veteranos já conheciam.

Foram demonstradas as funcionalidades, vistos os pontos altos e os baixos e foram detalhadas as funções biblioteconômicas dos módulos. Em entrevistas, alguns funcionários do setor comercial declaram que estas eram complexas e muito detalhadas, e que não viam necessidade da participação deles no processo, que essas eram as funcionalidades que o desenvolvedor deveria dominar e não eles, pois se tratava de construção de um produto. Porém os Desenvolvedores também acharam as explicações complexas e com muitos detalhes.

À época, a gestora questionou se ninguém iria anotar nada e a resposta foi *"temos livre acesso ao SofTeka e na hora de escrever as funcionalidades a gente abre".* Detalhes como Inserir, Alterar, Excluir, Ordenar, e as regras de negócio foram debatidas com os Desenvolvedores *in loco* além de receberem por e-mail as regras com os parâmetros básicos para a escrita do software.

A gestora tem um perfil detalhista e as apresentações eram feitas com muitos detalhes. Os Desenvolvedores reclamavam do nível de detalhes e da existência de que ela, *"sempre explicava o fazer biblioteconômico de cada coisa".* Com os princípios considerados básicos pela gestão para a escrita do sistema e com as informações do sistema espelho em mãos, *"era hora de fazer acontecer o projeto"* (Gestora).

O setor de desenvolvimento passou a interagir entre si para a escrita dos códigos relativos aos requisitos expostos. Em alguns momentos a equipe procurava a gestão com ideias para implementações diferenciadas a maioria das quais foram acatadas e incluídas no escopo do projeto. Dúvidas também eram dirimidas tão logo levantadas pelos Desenvolvedores. Regras de negócio eram debatidas com os bibliotecários da empresa que estavam à disposição dos Desenvolvedores.

A gestão tinha, continuamente, ideias de elementos a acrescentar no sistema, "*os desenvolvedores também traziam bastantes contribuições para melhorias e o sistema foi tomando uma dimensão com funcionalidades diferenciadas e inéditas para esse tipo de software"* (Gestora).

O conversar e depois codificar era uma rotina do desenvolvimento. Porém esta rotina ainda não supria as expectativas e demandas, nem da gestão, nem do setor de DS. Nesse período, também foi percebido que as chamadas "entregas regulares", prometidas durante as conversas de levantamento de informações para o desenvolvimento do sistema, não estavam sendo cumpridas.

O Engenheiro de Software, juntamente com o Coordenador do DS, entraram em contato com a gestão. Na opinião deles, era necessário se ter ajuda para

programar *"mais coisas"* e, até mesmo, ter a confiança se o que já havia sido codificado estava correto e em qual o grau de suficiência e eficiência.

Eles também ponderaram que o número de Desenvolvedores estava aumentando e era necessário maior controle das tarefas acontecidas no setor. Dever-se-ia achar uma "*metodologia em que o setor de DS recebesse o trabalho já devidamente destrinchado, aprovado pela gestão e validado pelos especialistas de domínio*" (Engenheiro de Software).

4.5 Criando softwares artesanalmente e o salto para a utilização do MPS.Br

As coisas estavam andando rápido na empresa e todos tinham expectativas específicas ao pensar em usos de metodologias de produtividade no setor de DS. Isso ia desde expectativa aumento nos lucros e criação de novos mercados pela empresa e de melhoria das relações de trabalho e empregabilidade pelos programadores.

Por ser uma empresa criada e concebida por Analistas de Sistemas e Engenheiro, suas atividades de construção e DS utilizavam conceitos de organização e metodologias de trabalho e escritas de códigos próprios à década de 90 e voltados para empresas de pequeno porte, próprios aos conceitos da época em que foi instituída.

Em 1999 o DS da maioria dos países em desenvolvimento era caracterizado pelas grandes dificuldades de acesso de pequenas empresas a soluções de ponta. O Brasil era considerado um país importador de tecnologias o que encarecia sobremaneira os valores finais de produtos desenvolvidos. O mercado profissional brasileiro estava começando a se consolidar na área de DS, situação ainda em andamento devido aos poucos investimentos em políticas públicas para o fomento à ciência e tecnologia nacional.

Mesmo assim, e atuando dentro desse contexto, a empresa buscava trabalhar com processos documentados e também procurava se aliar aos clientes, que geralmente possuem profissionais qualificados em suas funções, para criar produtos personalizados e adequados a cada realidade apresentada. Por esse motivo, era muito comum na organização que clientes, mesmo tendo escopos similares no mercado, possuíssem cada um o seu produto, desenvolvido com atores diferentes e, por conseguinte, com formatos, comandos e códigos diferentes entre si.

De acordo com a atual gestora, *"essa postura não rendia grandes valores de retorno financeiro, porém rendia à empresa um alto grau de fidelização e bom grau de captação de clientes em um mercado tão competitivo".* O grau de cancelamentos de contratos é muito pequeno e a empresa possui relacionamento direto com os gestores das empresas clientes o que facilita a melhoria constante dos sistemas utilizados. É muito comum a troca de ideias entre Desenvolvedores e setores de Tecnologia externos. A *empresa "é reconhecida no mercado pelo seu caráter relacional tanto interno quanto externo"* (Gestora), *"com pouquíssima rotatividade de clientes e funcionários"* (Diretoria Financeira).

Assim, até o ano de 2013, as relações de produção se davam de forma artesanal e muito próxima entre setores, gestores e empresas parceiras. Os Desenvolvedores (já em quatro) tinham ampla autonomia para decisões sobre o desenvolvimento dos produtos, além de incentivo para circular, telefonar, enviar e-mails e se relacionar diretamente com os atores da construção do software. De acordo com o Engenheiro de Software, era também muito comum que os desenvolvedores, gestores e demais colaboradores se reunissem informalmente fora do ambiente da empresa para falarem do mercado tecnológico no país e no mundo e tendências futuras para projetos e produtos.

No entanto, o fato de o setor de DS funcionar a partir das ações organizadas pelo próprio grupo era uma das reclamações da gestão, pois a documentação dos produtos ali desenvolvidos era, na opinião da gestão, bastante básica e a rastreabilidade das ações ali desenvolvidas muito elementares.

Mesmo insatisfeita com a forma como a gestão de DS era feita, a empresa seguia uma trajetória de credibilidade e ascensão em diversos setores de atendimento com produtos diversificados nas áreas de manutenção predial, gestão educacional e na Biblioteconomia. Esse crescimento estimulou os sócios da empresa a criar seus produtos futuros já em uma proposta de trabalho reconhecida pelo mercado do desenvolvimento de softwares.

Para dar concretude a esse projeto, os sócios e gestores se reuniram com toda a equipe de TI para buscar "*ferramentas que facilitassem o trabalho do DS ao mesmo tempo em que documentasse os produto*s" (Gestora). Um dos colaboradores sugeriu, então, a utilização de Metodologia Ágil SCRUM. Ele havia tido uma vivência em outra empresa com esta metodologia. Ele também havia conversado com a equipe de DS e *"a maioria conhecia via estudos na faculdade ou já haviam trabalhado com o método"* (Desenvolvedor).

Era março de 2013 e a gestão acolheu a proposta mesmo sendo informada que *"a documentação não era o "carro chefe" da metodologia, mas que poderia ser um primeiro passo para um DS mais visível",* (Desenvolvedor). Com a decisão já tomada, iniciou-se o "processo" de implementação. As novas ideias foram comunicadas a todos os Desenvolvedores e colaboradores e a metodologia SCRUM e seu jargão integraram-se rapidamente ao dia a dia da empresa se consolidando ainda no primeiro semestre de 2013.

Ainda assim, as insatisfações com as questões de documentação e rastreabilidade continuavam e, mesmo percebendo que o setor de DS estava trabalhando de uma forma mais organizada, a gestão buscava mais desempenho em relação às "*boas práticas de DS"*. Essa expectativa foi repassada ao Engenheiro de Software que se reuniu com a equipe para pensar possibilidades.

Pensou-se em consultores, até que foi sugerido pela equipe de DS, utilizar e integrar o MODELO DE MELHORIA DE PROCESSO DO SOFTWARE BRASILEIRO, o MPS.Br, ao uso da metodologia Ágil. A equipe sabia que era uma integração a ser acompanhada, por se tratar de mesclar metodologias com princípios diferentes entre si, mas, ao mesmo tempo encontraram na literatura

vários casos com esse tipo de integração e se sentiram *"tranquilos para propor e testar essa possibilidade"* (Engenheiro de Software).

A ideia era atender de uma forma mais eficaz às demandas da gestão em relação à documentação bem como atender a uma *"expectativa dos desenvolvedores de que a gestão e especialistas de domínio do produto de gerenciamento de bibliotecas também se envolvessem com a metodologia para produzir todos os requisitos e regras de negócios necessárias à codificação antes de o projeto iniciar, para que não houvesse retrabalho"* (Engenheiro de Software).

Os colaboradores do setor de DS sugeriram, também, que a empresa obtivesse uma certificação oficial do uso do modelo. A ideia foi acatada pela gestão como muito interessante uma vez que os parceiros que começavam a fazer parte da carta de clientes da empresa consideram que certificação em processos e métodos reconhecidos na área são valores agregados à empresa e além de ser uma certificação considerável para aproximar futuros clientes.

Dessa forma em outubro de 2013 o modelo escolhido: o MPS.Br, foi aprovado por toda a gestão e quadro societário da empresa. *"Os desenvolvedores estudaram o modelo MPS.Br a ser aplicado para sugerir à empresa, mas até agora nenhum tinha participado diretamente da aplicação dele"* (Engenheiro de Software). Através de pesquisa na internet a empresa encontrou a FUMSOFT, uma organização com sede em Minas Gerais e credenciada para acompanhar as empresas interessadas em possuir a certificação de Melhoria do Produto do Software Brasileiro e em fazer a implantação desta metodologia.

Em novembro de 2013 a empresa entrou em contato com a FUMSOFT, e uma reunião foi marcada com um representante da organização para a apresentação do modelo e uma foi feita uma explanação de como é executado o trabalho de implantação da MPS.Br e qual nível de maturidade a empresa pleitearia em um primeiro momento. O fato de a FUMSOFT estar sediada em Belo Horizonte/MG foi um fator muito decisivo para a empresa que tinha que pensar em nos custos de uma empreitada desse porte para o caso de pequenas empresas.

Nesta reunião, a empresa teve conhecimento de que a FUMSOFT iria criar o GRUPO 11, que se iniciaria em 2014 e que ainda havia vagas disponíveis para a entrada de empresas nesse grupo. De acordo com a gestora, a *reunião despertou em todos uma grande empolgação, pois, a princípio, a empresa tinha achado tudo que esperava em questões de organização e metodologias para o desenvolvimento de software".*

Assim, a empresa candidatou-se a uma vaga para participar do Grupo 11, fazendo a opção para o Nível G de maturidade. A FUMSOFT fez um diagnóstico formal das condições da organização, pois era necessário saber se a empresa estava apta a pleitear o nível G. A empresa foi aceita no dia 20 de novembro de 2013. Já em dezembro de 2013 assinou o contrato para a implantação do MPS.Br como modelo de gerenciamento de projetos e requisitos de produção de softwares.

Passado esse momento, o início dos trabalhos se deu em fevereiro de 2014. Os gestores, o Coordenador do DS e o Engenheiro de Software da empresa pesquisada participaram de uma reunião inicial do Grupo 11, nos escritórios da FUMSOFT, onde a consultora que iria acompanhar o processo foi apresentada aos representantes da empresa. Em 18 de fevereiro de 2014 aconteceu, oficialmente no escritório da contratante, a primeira reunião com os Desenvolvedores para a implantação do MPS.Br.

A empresa crescia em demandas de DS, e, até o momento, esse parecia ser o passo correto e necessário a se dar para que os processos continuassem seguindo com qualidade. As reuniões com a FUMSOFT foram acolhidas com interesse e todos entendiam que a proposta para o novo software de bibliotecas era a oportunidade para a aplicação do modelo e inicio de *"uma nova era institucional"* (Diretora Financeira).

Nos próximos itens acompanharemos o desenvolvimento do novo software de bibliotecas pelo seu desafio de escrita particularizada de códigos, por sua

peculiaridade no mercado e pelas controvérsias geradas desde sua concepção dentro do modelo.

4.6 Expectativas e medos relativos ao MPS.Br

Sistemas gerenciadores de acervos para bibliotecas possuem muitas regras e padrões de negócio. Reescrever e integrar um software já complexo a uma nova forma de pensar os serviços oferecidos por bibliotecas também pareceu um grande desafio para o grupo que ainda estava estudando e implantando as novas ferramentas utilizadas para construção do novo sistema. *"Para uma empreitada dessa magnitude, a empresa buscou formas de garantir uma maior qualidade e controle do desenvolvimento desse sistema*" (Gestora).

Era abril de 2014 e a fase de diagnóstico da empresa que acompanhou a preparação para a certificação MPS.Br já havia sido concluída. Os gestores estavam muito satisfeitos com o material levantado e com a postura profissional da consultoria. *"Os materiais eram muito claros e bem organizados. Os desenvolvedores estavam esperançosos e contribuíam para a implantação do novo modelo"* (Especialista de Domínio). O processo estava sendo bem avaliado por todos e parecia que tudo corria de acordo com o esperado. *"É muito bom saber que a empresa optou por ter a certificação, ela é muito reconhecida, é bom para o currículo da gente passar por uma implementação de certificação desse porte"* (Desenvolvedor).

Uma implantação feita de forma correta do modelo MPS.Br, era muito importante naquele momento uma vez que o uso de um modelo com esses padrões, em uma estrutura de pequena empresa, pode ser bem mais árduo que para as grandes, devido à ampla necessidade de investimentos, treinamentos e recursos, nem sempre disponíveis nas menores. Em relação a essa importância, o Engenheiro de Software expressava a mesma opinião, acrescido do fato que ele considerava que o empenho seria grande, mas o modelo trazia uma perspectiva considerável para a corresponsabilidade.

No modelo MPS.Br, são 19 metas para se chegar ao gerenciamento de projetos e 25 para cumprir as metas do gerenciamento de requisitos. Isso para se chegar ao nível G de maturidade. Essas exigências implicaram em aumento dos esforços da equipe e também de toda empresa. Todos tiveram de se adequar à nova proposta através de estudos, leituras, treinamentos, redesenho de espaços, inclusão de novos sistemas informatizados de controle da informação entre outros.

De uma certa forma, todo esse movimento interno era previsto e a equipe lembrou do fato que deveriam implementar o MPS.Br aderindo-o à metodologia SCRUM, já utilizada na empresa. Era importante que houvesse o entendimento que o MPS.Br pedia uma forma mais rígida de gerenciamento de projetos e requisitos e nesse momento a gestão não abria mão disso. Parte do sistema de bibliotecas já estava pronto, e, para que a aplicação do modelo começasse, uma primeira tarefa no formato MPS.Br foi iniciada: a construção do *"módulo de cadastro de livros"*.

Mesmo a equipe estando motivada para o processo, era um momento muito novo para todos, o que, na opinião do Engenheiro de Software, gerou algumas tensões no grupo e também distanciamentos, uma vez que os ocupantes dos cargos de Engenheiro de Software e Coordenador do setor tiveram de fazer valer sua autoridade hierárquica perante os grupos para garantir que os protocolos do MPS.Br fossem observados.

Assim, uma das primeiras ações do setor de DS após iniciarem mais profundamente os estudos para a implantação do modelo, foi o de comunicar à Gestora que, a partir daquele momento, e seguindo o guia MPS.Br, as formas como as demandas da Gestão e Especialista de Domínio chegavam ao DS para novos requisitos, solicitações de mudanças de escopo e priorizações deveria mudar.

De acordo a equipe, essas solicitações deveriam ser discutidas com antecedência em reuniões com um grupo formal que decidiria se as mudanças eram mesmo necessárias. Somente partir dessa reunião o Engenheiro de Software poderia

introduzir no processo as demandas para o sistema informatizado de controle de tarefas. Somente seriam essas tarefas executadas pelos Desenvolvedores em pequenas partes e, depois, seriam reunidas no sistema global com entregas a cada 15 dias (tempo do ciclo de cada *sprint*).

Mesmo com receios em relação a como isso se daria internamente, a gestão aceitou a nova regra e também foi informada que qualquer pergunta, duvida ou nova demanda devia passar primeiramente para o Engenheiro de Software, o qual seria o elo com os Desenvolvedores. O clima era de expectativa.

5 AS DORES DO PARTO: CONFLITOS NA IMPLEMENTAÇÃO DO MPS.BR

A implementação do MPS.Br exige o atendimento de várias premissas que vão desde o atendimento para o Gerenciamento de Requisitos (GRE), ao atendimento para o Gerenciamento de Projetos (GPR). De acordo com o Guia Geral MPS de Software, publicado pela SOFTEX, (2016), o objetivo do modelo, a longo prazo, é dar subsídios para que metas técnicas e metas de negócio sejam aplicadas ao longo do processo de forma a se obter o resultado final de melhoria do processo de DS necessários para a sustentabilidade das empresas de desenvolvimento de software no mercado brasileiro.

Em relação à empresa estudada, o início da aplicação do modelo MPS.Br já havia acontecido e, agora gestores e equipe de DS estavam se adaptando a detalhes de implantação. No princípio do processo se pensava que a contratação de dois Desenvolvedores era suficiente para levar a cabo a construção do software. Porém, já no começo do processo o Engenheiro de Software pediu nova contratação.

5.1 De 2014 a 2015 e o SISTEMA?

Em maio de 2014 foi contratado mais um desenvolvedor, somando agora três Desenvolvedores dedicados a esse projeto. Esse desenvolvedor já veio com total domínio das funcionalidades, limitações, e coisas consideradas de cunho positivo do software espelho, o SofTeka e que deveriam ser integradas ao YesTeka.

Nesse ínterim, ainda em maio de 2014 os sócios da empresa, decidiram mudar o escritório para um novo espaço físico de uma sala de 60 m² para um andar de 340 m². Esta decisão fazia parte do planejamento estratégico da empresa que desejava:

> Ter mais desenvolvedores, fomentar a existência de grupos de práticas em bibliotecas para fazer um produto surpreendente. Fazer apresentações do sistema novo YesTeka. Buscar parceria com Universidades e Escolas de Biblioteconomia para ampliar conhecimentos. Ter um espaço maior destinado a prestação de serviços de automação de bibliotecas caso a modelagem da MPS.Br demandasse, ter espaço para contratação mais de pessoas. Ter espaço maior e melhor para os desenvolvedores e para a instalação de duas lousas eletrônicas: uma exclusiva para o desenvolvimento e outra para o auditório; Agregar mais conforto e beleza à empresa (E-mail do setor de comunicação para os colaboradores).

Também durante tal processo, a empresa teve alguns problemas com a mudança de sede, cabeamento, telefonia, suporte, desvios de funções etc., tudo isso em meio à construção de um novo produto e implantação de um novo modelo de trabalho.

Em março de 2014 a empresa já se encontrava dentro de *"um certo padrão de normalidade"* (Engenheiro de Software). A gestão esperava uma entrega que não aconteceu, a justificativa da equipe de DS foi que não chegaram ao objetivo devido aos problemas da mudança para a nova sede. Novo prazo foi negociado para abril de 2014.

Em abril foram entregues os seguintes módulos:

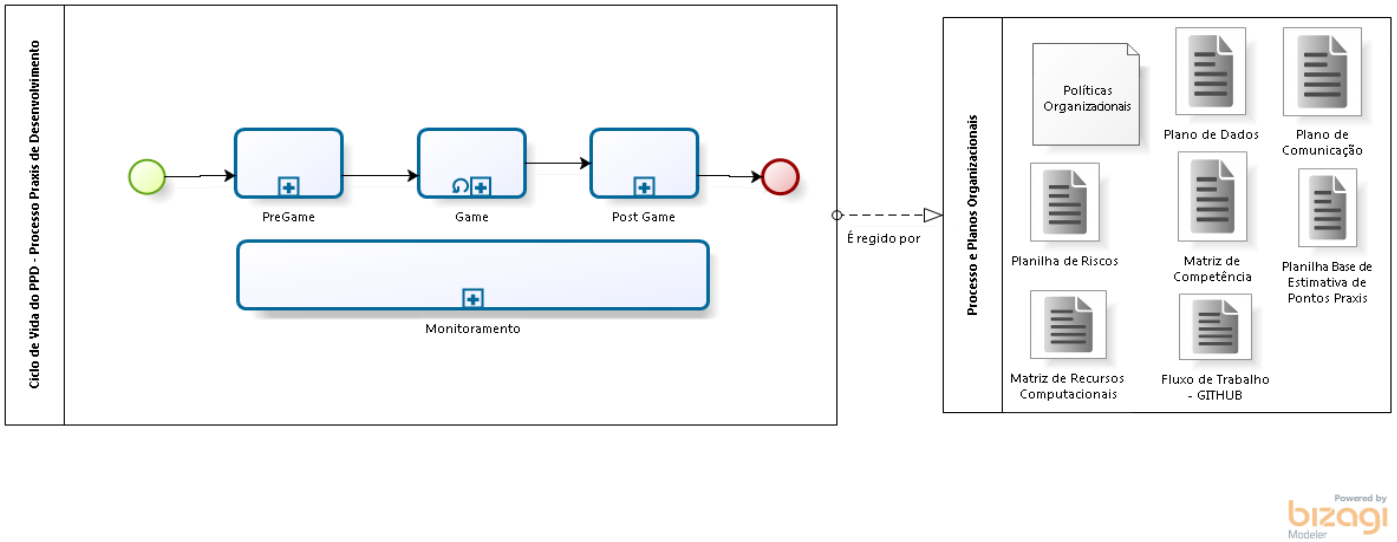
a) Cadastros Auxiliares
b) Cadastro de Usuários
c) Parâmetros do Sistema
d) Parâmetros de Usuários

Em julho de 2014 o MPS.Br já estava com os sistemas de controle de tarefas implantados (Uso de uma ferramenta de controle chamada Redmine e outra chamada KanBan) e também com alguns modelos de relatórios prontos e disponibilizados na rede. Os Desenvolvedores já estavam usando o jargão do modelo e iniciavam-se os testes para que um projeto rodasse dentro do modelo inserido na sua totalidade.

Para que a empresa fosse certificada, os auditores deveriam ter acesso pelo menos dois projetos rodados respeitando todas as 44 metas do modelo e outro

projeto rodando dentro do modelo. O processo deveria estar todo documentado através do Plano de Desenvolvimento da Empresa e disponível a toda a equipe de DS. Vide FIG. 5.

FIGURA 5 - PLANO DE DESENVOLVIMENTO DA EMPRESA - MPS.BR



Fonte: Documentos cedidos pela empresa.

5.2 Primeiro grande conflito

Um dos projetos que fariam parte desse teste para rodar um projeto usando o modelo MPS.Br, era o do modulo de "Cadastro de Livros". A empresa contava com três profissionais especialistas na área biblioteconômica, um encarregado diretamente a este projeto atuando como Especialista de Domínio.

Na opinião da gestora essa prerrogativa unida ao uso do modelo MPS.Br prometiam entregas mais padronizadas e dinâmicas. Porém, de acordo com as profissionais sobre a primeira entrega de um módulo completo: *"a decepção não poderia ser maior"*. "Funcionalidades pela metade, inconsistências, poucas ordenações alfabéticas ou numéricas, hierarquias dos assuntos da

biblioteconomia sem consistência, entre outros problemas, além de telas que não tinham fundo" (Especialista de Domínio).

O módulo de "Cadastro de livros" apresentou um erro conceitual e teve de ser refeito na sua totalidade, só as telas seriam aproveitadas. O módulo demorou três meses para ser entregue, houve vários momentos em que os Desenvolvedores acessavam as Bibliotecárias para dirimirem dúvidas. Todas as entregas feitas para estas durante o período foram aprovadas.

Os requisitos escritos pelas Bibliotecárias para o desenho da página de cadastro de livros previam conceitos biblioteconômicos específicos e inegociáveis para a construção do módulo, mas quando o módulo foi entregue, ele apresentou uma descaracterização do primeiro conceito colocado como requisito (Obra é igual a conjunção dos campos título, título original, autoridades e ISBN). Esta descaracterização inviabilizou todo o trabalho realizado pela equipe no período. "*Porém, foi apresentado um protótipo e esse protótipo foi aprovado pelas bibliotecárias"* (Desenvolvedor).

Aconteceu, nesse momento, o primeiro grande conflito. Os Desenvolvedores culpavam o gerente de TI e afirmavam que os requisitos não possuíam especificações e regras de negócio suficientes para um resultado satisfatório. A gestão não aceitava a entrega, pois, duas regras básicas não foram obedecidas:

> **1. Não se pode construir nada sem antes consultar o SofTeka.**
> **2. Em biblioteca tudo se ordena alfabeticamente e numericamente.**

Estas regras haviam sido enviadas para todos, via e-mail, em documento de comunicação da gestão, em caixa alta, como compilado acima além de os Desenvolvedores terem participado de um treinamento no sistema anterior. Foi feita então a impressão de todas as telas do SofTeka destes referidos módulos, e uma analista da área de desenvolvimento foi alocada no projeto YesTeka Bibliotecas para listar e levar em forma de listas as funcionalidades descritas, juntamente com as melhorias sugeridas.

Já era outubro de 2014 e as expectativas de certificação no MPS.Br eram para março de 2015. E o primeiro acompanhamento utilizando parte da metodologia mostrou que a qualidade da entrega não estava como a esperada pela equipe. O Engenheiro de Software viu como positiva a constatação, *"por ter sido feita no início e não quando o projeto fosse todo entregue, no final",* a gestão viu com estranheza tantas inconsistências e atribuiu o fato *"a se estar fazendo dois processos complexos ao mesmo tempo"* e buscou recursos *"para ajudar o Engenheiro de Software na organização das metodologias e processos requeridos pelo MPS.Br".*

Para atender à presente demanda, começou-se um processo de seleção de pessoal. A gestão da empresa optou por investir em um profissional mais antigo na casa. Foi escolhido um dos Desenvolvedores do sistema SofTeka, porém as linguagens eram diferentes e ele tinha dificuldade de tramitar nas ferramentas utilizadas para programar em web. A empresa resolve, então, pagar uma pós-graduação para o funcionário, pois, em princípio, ele era o desenvolvedor com mais conhecimento das regras de negócios do software anterior.

Este investimento não deu certo. Após o primeiro bimestre, o funcionário devolveu o dinheiro investido no curso e disse que não iria continuar os estudos. Foi contratada uma empresa para que fizessem a seleção, mas ninguém foi selecionado. Por indicação, apareceram dois candidatos: um da Bahia e outro Colombiano. Foram selecionados. Agora eram 4 Desenvolvedores dedicados à escrita do Código do Sistema de Bibliotecas.

A sala foi ampliada, o Engenheiro de Software mudou de sala para ficar junto da equipe, porém as entregas continuaram com alguns problemas. Ao intuir que o sistema estava com muito atraso, os gestores da empresa resolveram aumentar a equipe de vendas para buscar recursos nos outros produtos. As reuniões para definir os próximos ciclos do software continuavam a acontecer no mesmo formato. Havia agora um novo fator, a dificuldade linguística e de entendimento do desenvolvedor colombiano. Embora o mesmo falasse português nem sempre o contexto das situações mencionadas ou descritas acabava por não fazer parte

das experiências de vida dele e tornava o entendimento do que foi pedido mais mais complexo.

A nova equipe continuava com dificuldades com a qualidade *versus* Produtividade *versus* tempo *versus* entrega *versus* prazo *versus* necessidades urgentes da empresa em ter um produto minimamente vendável. Novas pessoas foram contratadas e o quadro de funcionários teve um aumento significativo *"tudo em função da manutenção da construção do* YesTeka*"* (Diretora Financeira).

Para dar suporte à implantação do MPS.Br foi contratado também, um Gerente de Projetos (GP) que, juntamente com o Engenheiro de Software e o Coordenador da área de DS reorganizaram o modo de trabalho como era feito até o momento. Novamente, foi requisitado formalmente que nem as gestoras nem os especialistas de domínio enviassem ou discutissem requisitos diretamente com os Desenvolvedores.

Agora eram três chefias que afirmavam que os programadores deveriam se dirigir a elas para dirimir "dúvidas técnicas". De acordo com eles, (Gerente de Projetos, Engenheiro de Software e o Coordenador da área de DS), se a dinâmica de trocas de informações não ficasse centralizada neles, a construção do software no prazo prescrito seria impossível. "M*esmo porque a gestora e a analista de domínio entravam no setor de DS para pedir inclusões, mudanças e perguntar sobre como o projeto estava, acabando por distrair o grupo, que tinha escopo e metas já delineadas"*(Coordenador DS).

Em princípio não se podia mais entrar no setor. Agora os Desenvolvedores deveriam dirimir dúvidas dentro do DS. Continuou-se com receios, mas o pedido foi acatado pela gestão. Assim, os grupos foram orientados a se comunicar somente nos padrões adotados. Nesse novo modelo, cada programador tinha contato apenas com pequenas tarefas que faziam parte de um "todo", *"que se pensava estar sendo documentado devido às prerrogativas do MPS.Br, que mesmo estando em processo de implantação, previa*

documentação dos gerenciamentos de requisitos e de projetos". (Gestora da empresa).

A gestão aceitou a formalização de distanciamentos, *"mas foram reforçados os prazos de entregas quinzenais"*. Porém *"a equipe não conseguia fazer entregas dentro de qualquer cronograma que fosse possível"* (Gestora). A frustração tomou conta da equipe de Desenvolvedores, dos gestores e o setor administrativo financeiro se manifestou sobre a provisão orçamentária para esse projeto já ter ultrapassado os valores previstos. Houve um grande "mal-estar" devido a esse episódio.

5.3 Segundo grande conflito

O segundo grande conflito veio emendado a esta situação de mal-estar. Chega o mês de dezembro de 2014. A MPS.Br estava sendo implantada e não havia nenhum tipo de reclamação ou maiores conflitos em relação ao atendimento da consultoria. Porém a construção do sistema em paralelo com a implantação trazia uma *"certa angústia"* a todos.

Os Desenvolvedores se adaptando às rotinas de Gerenciamento de Projetos e Gerenciamento de Requisitos do MPS.Br, a gestão preocupada com os prazos devido ao atraso na entrega seguido da refatoração do módulo de Cadastro de Livros e a diretoria financeira preocupada com as novas contratações versus entrega do sistema vendável para manutenção dessa nova equipe.

Com a ajuda da consultora, do recém-chegado Gerente de Projetos aliado ao Engenheiro de Software, Especialista de Domínio e também da Engenheira que atendia ao software de Manutenção os dados começaram a ser inseridos e acompanhados pelas telas do sistema de controle (Redmine). Ali começaram a se configurar diversas entradas de dados que ajudavam no acompanhamento das tarefas pelo Gerente de Projetos e também em dados para relatórios com informações mais concretas sobre em quais estágios de construções cada tarefa estava.

Nenhum funcionário, por uma determinação oficial da gestão, estava autorizado a entrar, perguntar, requisitar ou mesmo telefonar para o setor de DS sem antes passar ou pelo ES ou pelo Gerente de Projetos. Agora na metodologia a equipe se comprometia formalmente ao fim de cada reunião de validação de novo projeto a cumprir com os prazos votados por eles. Eles assinavam, mas ao final de cada projeto muitas tarefas ficavam para trás e o novo ciclo do projeto *(sprint),* já vinha com as tarefas atrasadas do anterior.

No terceiro projeto, a gestão reclamou com o recém-chegado Gerente de Projetos que não entendia porque havia tantos atrasos de entregas, já que os Desenvolvedores estavam com os requisitos na pilha e que ninguém havia pedido inclusões de tarefas nem mudanças de escopo. O setor de DS, na pessoa do Gerente de Projetos passou a ter reuniões diárias com a gestora e mensalmente com representantes da empresa organizado para atender a uma das metas do MPS.Br. A ideia era alinhar com a alta gestão da empresa as necessidades e dificuldades do setor.

Havia um protocolo formal de monitoramento. Para acompanhar as ações de Gerenciamento de Projetos e Gerenciamento de Requisitos, o Gerente de Projetos também fazia reuniões diárias no setor de DS. Essas reuniões deveriam ser de 15 minutos se seguissem a Metodologia Ágil (já aderente às documentações do MPS.Br) que estava em processo de aderência ao MPS.Br. No entanto, essas reuniões começaram a durar mais de uma hora e os Desenvolvedores verbalizaram para o ES que não estava sendo produtivo. Eles tinham de desenvolver códigos, se apropriar das horas relativas à realização das tarefas e participar de reuniões. Junto a isso eles percebiam a insatisfação da gestão nas reuniões de entrega.

Devido ao fato dos desenvolveres codificarem pequenas tarefas, não era mais possível acompanhar o crescimento dos módulos "a olhos vistos", como acontecia anteriormente. Foi quando houve mais um grande conflito, o corpo administrativo da empresa "*queria e entendia que era preciso registrar e*

documentar" (Gestora), mas a gestora estava achando o processo lento, burocrático e engessado.

As reuniões eram demoradas e o corpo administrativo inferiu que tantas reuniões *"comiam"* o tempo de desenvolvimento, comprometendo a produtividade do DS. Esse sentimento foi relatado em reunião administrava à Gestora e aos sócios. *"A sensação era a de que os desenvolvedores se tornaram tarefeiros e as trocas, criatividade, as inovações, as boas ideias pararam de acontecer"* (Diretora Financeira).

As tarefas eram dispostas pelo Engenheiro de Software em forma de tarefas em uma plataforma chamada Redmine. Os Desenvolvedores tinham de fazer as entregas das tarefas descritas na ferramenta Redmine, e, depois, tinham de registrar as horas via sistema para o acompanhamento da Gerência de Projetos. Essa rotina parecia pesada e o modelo MPS.Br ainda não estava acontecendo em sua totalidade. A gestão expressou isso em reuniões, mas o processo estava em conformidade com os padrões do modelo estabelecido e "*isso poderia ser uma reação momentânea e adaptativa do grupo"* (Engenheiro de Software).

Os tempos para a execução das tarefas começaram a ser questionados pela gestora. Nem sempre a relação de tempo *versus* esforço pareciam lógicas para ela. As entregas continuaram sendo feitas com problemas e *"a equipe entrou em um processo de refazer, refazer."* (Gestora). Até esse momento ninguém tinha verificado, testado ou analisado a qualidade dos códigos desenvolvidos. As horas extras continuavam durante a semana e nos finais de semana.

Mesmo com as entregas atrasadas a gestão não interferia no processo de escrita e criação do código, mas cobrava do Engenheiro de Software e do Gerente de Projetos posicionamentos e sempre colocava em reuniões sua insatisfação com os rumos que o DS estava tomando. Passou-se a testar as entregas. Algumas entregas consistentes foram feitas, outras tantas com os mesmos tipos de falhas detectadas na codificação do módulo de cadastro de livros.

O Coordenador tinha se afastado do processo de implantação do MPS.Br. Ele havia sido um dos principais entusiastas da aplicação do modelo ajudando a criar regras, tabelas, o Plano de DS da empresa entre outros. Com a chegada do GP que veio para ajudar na implantação do MPS.Br e com os requisitos nas mãos do ES, ele passou a desenvolver junto com a equipe.

Em fevereiro de 2015 o Coordenador pediu demissão. Ele vinha com grande número de atestados, havia feito uma cirurgia grande já estava afastado há mais de um mês. Este alegou que não estava produtivo e que sabia que a empresa precisava de produtividade naquele momento. Foi um dos colaboradores que trouxe muitas ideias e inovações para vários projetos desenvolvidos e era um profissional de inteira confiança da casa.

Porém, de acordo com a gestão e com os próprios colegas de trabalho ele realmente não estava satisfeito com a produtividade do grupo e também expressou a eles que estava se sentindo muito subutilizado no setor. Em entrevista à pesquisadora ele não trouxe estas referências, mas destacou que era necessário dar agilidade aos trabalhos desenvolvidos.

2015 era o prazo estimado pelo setor de DS para se ter um produto com as principais funcionalidades implantadas e testadas para venda. Todas essas situações ocorrendo ao mesmo tempo em que o prazo e o dinheiro captado no mercado não estavam dando retorno. Isso acabou por desencadear uma ruptura drástica na forma como as coisas estavam acontecendo no setor.

5.4 Ruptura: Construção conjunta do módulo de Circulação de materiais.

Em uma ferramenta de controle de acervos de bibliotecas, um dos módulos mais utilizados é o de Circulação de Materiais. O Especialista de Domínio tinha uma grande preocupação com este módulo. Ele requer muitas regras de negócios para que se tenha total controle do acervo e do histórico do aluno na biblioteca.

Esse módulo tinha de ser muito bem amarrado e, desde que a empresa optou por usar o modelo de MPS.Br que "*os erros mais grotescos"*, de acordo com a gestão, haviam acontecido.*"Estamos com problemas na plataforma Zend, ela dificulta a evolução do sistema e foi descontinuada antes do sistema ter começado a ser construído"* (Desenvolvedor). *"Vi ali um pedido de socorro implícito, hora de procurar o Engenheiro de Software"* (Gestora).

Como foi narrado anteriormente, antes da implantação dos modelos, os especialistas de domínio da empresa discutiam os requisitos diretamente com os Desenvolvedores e quando havia dúvidas eles também podiam procurar diretamente o especialista para dirimi-las. Com a nova metodologia já acontecendo e no seu terceiro projeto rodando no modelo os requisitos iam para o Engenheiro de Software que os traduzia de forma escrita em histórias e tarefas, as quais os programadores acessavam via uma ferramenta de gerenciamento de tarefas.

Quando os Desenvolvedores tinham dúvidas, eles entravam em contato com o Engenheiro de Software que procurava os Especialistas de Domínio para entender a especificação e reescrever as tarefas, caso fosse necessário. Essa metodologia faz parte do modelo MPS.Br. Tudo isso lembrando que foi designado um Especialista de Domínio totalmente dedicado ao projeto do YesTeka.

Assim, seguindo o modelo, foi expresso em reunião com a gestão e representantes da empresa que "*os especialistas tinham de escrever o projeto completo do sistema desejado para introdução deste na plataforma de acompanhamento dos requisitos*" (Gestora). A gestão considerou que seria impossível para a estrutura da empresa escrever todos os requisitos de um sistema antes de iniciar o software. Ainda mais um projeto cujas funcionalidades estavam sendo ainda pesquisadas e divulgadas na academia e institutos de controle e normas da Biblioteconomia.

A gestão também se expressou e não entendia que esse era o modelo ideal para se construir softwares num mundo tão dinâmico e mesmo que o modelo

trouxesse essa prerrogativa *"a empresa via como impossível de se atender"*. Foram contratadas horas da consultora para agilizar os processos e discutir esse tipo de demanda de escrita completa dos requisitos do sistema.

A consultora se reuniu com a equipe e foi desenhado um processo onde o planejamento de um projeto seria feito três ciclos (*sprints*) antes do início do projeto e que não seria necessário se escrever e aprovar todos requisitos e regras de negócio antes de se iniciar um software e que o modelo MPS.Br aceitava essa prerrogativa. Porém, os atrasos e erros continuavam.

5.5 (des)Entendimentos

Através do vidro transparente a gestora vê, em pleno horário de trabalho, três funcionários da área de desenvolvimento fazendo compras de tênis pela internet. Ela chega, entra na sala de desenvolvimento e fica parada atrás deles *"observando".*

Há um constrangimento, sistema atrasado, entregas incompletas e compras no horário comercial. "*Eu vi aquilo e fiquei indignada, porém entendia que eram necessárias paradas no setor, pois o tipo de escrita que eles fazem é muito extenuante"* (Gestora). A novidade naquele momento era a Gestora, depois de meses sem entrar no setor estar ali.

Então, um funcionário diz: *"se eu tivesse que comprar um produto de cada vez e para comprar outro eu tivesse de fechar o procedimento, eu compraria muito menos"*. Ao ouvir essa fala do desenvolvedor a gestora chamou o Engenheiro de Software: *"Está aí a ideia: carrinhos site de compra: operações simultâneas sem perda de informação".*

Neste momento, houve uma completa quebra da barreira imposta pelo Engenheiro de Software, Gerente de Projetos, Desenvolvedores, pela norma da MPS.Br ou pelas práticas de gestão que estavam em implantação e também as que estavam em vigor na época. A gestora pediu para pararem o DS e disse o

que queria que fosse feito: um módulo de circulação de material com conceito de carrinhos de compras para operações de empréstimo, devolução, reservas e renovação simultâneas.

A gestora perguntou aos Desenvolvedores se eles achavam factível e, se sim, quanto tempo para se construir essa funcionalidade. A equipe disse que era factível, mas que não sabiam a estimativa. Ainda nessas discussões, um desenvolvedor disse que poderiam usar o conceito de *Work Space* para atender a essa demanda e a equipe estimou um mês, mas precisavam do Especialista de Domínio à disposição deles para acompanhar os protótipos. Os Desenvolvedores disseram também que precisavam entender a fundo como deve ser um bom módulo de circulação de materiais.

> Usualmente os sistemas de biblioteca fazem uma operação de cada vez: ou o operador empresta, ou devolve, ou renova, ou reserva o material. Caso um usuário "A" que esteja no processo de empréstimo, seu nome já esteja na tela e ele resolve buscar outro material, para que a fila ande, o operador tem que buscar os dados do usuário "B", atende-lo e na volta do usuário "A" tem que fazer novamente toda busca do nome deste usuário e do livro a ser emprestado (Gestora).

Assim, o "requisito" e as "regras de negócios" eram:

"Desejamos fazer as quatro operações conjuntas e que a busca do usuário se desse de uma única vez. Não havia conhecimento de um software de biblioteca com esta funcionalidade". Partiu-se, então, para verificações de outras possibilidades e pesquisas em modelos de site.

A partir daí, o Especialista de Domínio e os Bibliotecários voltaram a ter contato direto com os Desenvolvedores. Em um esforço conjunto das equipes dos Desenvolvedores foi construída uma lista com o que se pretendia e o *layout* da tela. *"Foi muito bom voltar a ver o desenvolvimento parecer novamente desenvolvimento. Tinha interação, tinha trocas. As bibliotecárias escreveram em um quadro: Quais são as regras para se construir um módulo de circulação de material mais inteligente e surpreendente?"* (Gestora).

A gestão também acompanhava esse movimento e *"percebia ali, que aqueles jovens voltaram do universo tarefeiro para pensar, criar inovar e que era só buscar o melhor deles".* Até então mesmo sendo uma imposição gerencial que se consultasse o SofTeka para iniciar projetos, isso não era acatado. Dessa vez os Desenvolvedores pediram para abrir o sistema para ver como era, e nesse caso, *"NÃO fazer igual, fazer diferente".*

A gestão considerou esse momento muito significativo. De acordo com ela houve produtividade, satisfação, criatividade e índice baixo de reconstrução. *"O tempo foi compatível, pois, sabíamos que eram muitas as regras de negócio. Só após dois dias de intenso estudo e trocas, depois que sabíamos de muitas coisas que teriam que ser contempladas neste módulo, que o protótipo foi construído"* (Desenvolvedor).

O protótipo foi apresentado com algumas poucas funcionalidades. Mas muitos Desenvolvedores chamavam a Gestora ou a Especialista de Domínio para *"verem"* as telas que eles estavam construindo em suas máquinas. Porém, diferentemente de outros momentos, as funcionalidades estavam completas o suficiente para o funcionamento básico do sistema. A entrega foi visualizada e testada e as equipes consideraram a ideia do uso dos carrinhos um sucesso. *"Todos estavam mais tranquilos e com a sensação de dever cumprido, foi cansativo, mas voltamos a trabalhar um com o outro. O desenvolvedor "A" me ajudou bastante e eu também ajudei"* (Desenvolvedor).

Essa situação aconteceu à revelia do Engenheiro de Software e do Gerente de Projetos no calor de uma situação quando a gestora entrou na sala e a equipe não estava envolvida na escrita do software. Tanto o Engenheiro de Software quanto o Gerente de Projetos estavam na sala quando aconteceu isso. Eles participaram do processo apenas acompanhando a equipe e tentando documentar algumas etapas do processo. Foi uma situação complicada, pois de certa forma, eles *"foram afastados"* ou se afastaram.

5.6 De volta aos métodos

Nem tudo são flores. Essa quebra de protocolos atrasou o processo de certificação que ficou praticamente parado durante dois meses. Passado o momento onde houve interferência drástica na aplicação da MPS.Br, a gestão determinou que os procedimentos da aplicação para certificação voltassem a acontecer.*"Era preciso voltar ao projeto de documentar o software, mesmo por que precisamos funcionar em padrões reconhecidos do mercado. Algumas licitações estão pedindo o certificado e já estamos na reta final para conquistá-lo"* (Gestora).

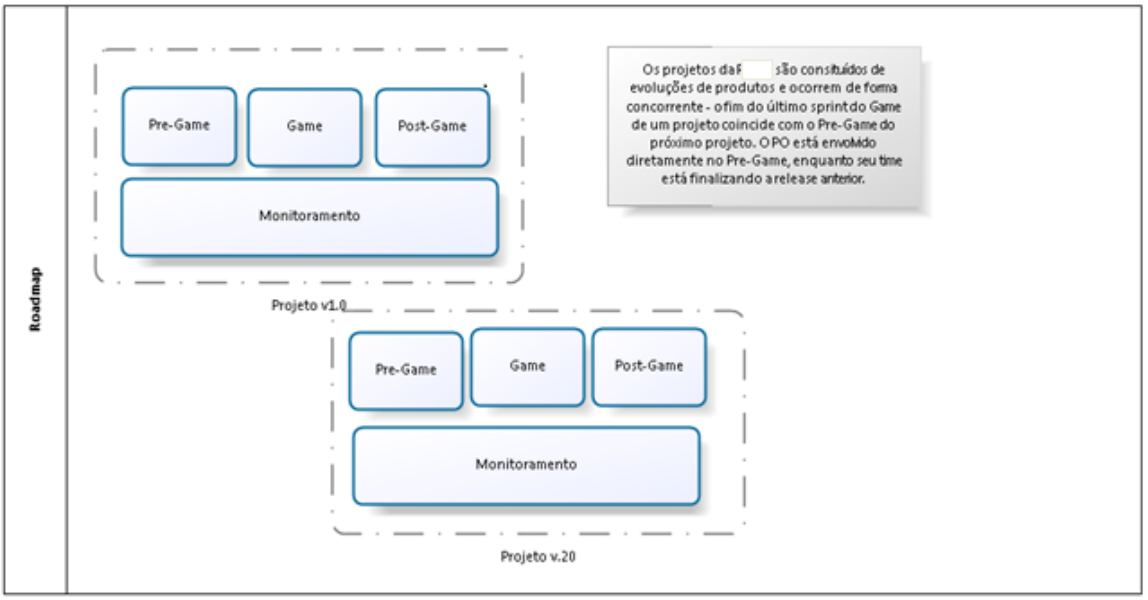
Mesmo com todos os conflitos, medos e desafios gerados na implantação, todos concordavam que a certificação era importante. Porém, os Desenvolvedores pediam flexibilizações. Por exemplo, eles queriam usar o método sem a rigidez das documentações e, agora, pediam requisitos melhores escritos. *"Voltamos à fábrica de requisitos, é mais difícil no começo, mas depois estabiliza"* (Desenvolvedor). Os Desenvolvedores não reclamavam que os requisitos chegavam em forma de pequenas tarefas, mas os gestores se sentiam incomodados por não verem entregas mais robustas.

Os Desenvolvedores insistiram que era necessário continuar investindo na criação de requisitos bons e justificaram dizendo que a qualidade do produto de software depende de um requisito bem escrito e completo e que os problemas anteriores vieram por causa de requisitos ruins. *"Requisitos têm de ser bem escritos e têm de chegar pra gente com todas as regras de negócio já pensadas e contidas nele"* (Desenvolvedor). *"Me mostra um requisito pra eu seguir"* (Especialista de Domínio). *"Para ficar bom temos de pensar ele em parceria, você tem de estar aqui comigo"* (Coordenador).

A gestão queria voltar ao modelo por causa da certificação, que é importante mercadologicamente e também porque não queria abrir mão da rastreabilidade, documentação e nem do protocolo formal de comprometimento com os prazos e estimativas de entregas que a equipe devia fazer. Havia uma distorção na

expectativa que a cada 15 dias deveria ficar pronto uma funcionalidade completa e "ver" o que foi feito era uma cobrança diária por parte da Especialista de Domínio e da Gestora a partir do acompanhamento da criação e evolução dos projetos no sistema informatizado, FIG 6.

FIGURA 6 - MAPA DE CRIAÇÃO DE PROJETOS DA EMPRESA



Fonte: Documento cedido pela empresa.

O tempo ia passando e os clientes do software antigo, o SofTeka, cobravam migrar para o YesTeka, que é em plataforma WEB. Já era fevereiro de 2015: funcionários sem férias, mas muito cansados da "tensão". Por esse motivo, foram feitas concessões para que um funcionário, que era colombiano tirasse alguns dias para ir para casa. O outro funcionário que era da Bahia também foi para casa, outros com férias vencidas também foram descansar. *"A vida só volta ao normal na segunda quinzena de março."* (Gestora Financeira).

Abril, maio e junho passaram com altos e baixos. Já haviam sido feitas seis entregas de projetos. De repente, tudo parece ficar muito mais lento e moroso. Nesse período houve mudanças de escopo, acréscimos de funcionalidades, reuniões para priorizações e, na primeira quinzena de julho, iniciaria o primeiro projeto que deveria ser rodado a partir da completude de atendimento das metas

do modelo MPS.Br. *"Havia um silêncio... As coisas não pareciam andar bem no DS"* (Especialista de Domínio).

O primeiro projeto oficial preparado já com vistas para a avaliação estava prestes a ser rodado. As ferramentas de controle já estavam implantadas (KanBan, Redmine e um diretório de armazenamento das documentações com modelos, políticas e princípios construídas durante o processo de implantação). Os Desenvolvedores registrando as horas usadas para cada tarefa pronta e a gestão validando priorizações junto ao Gerente de Projetos. No entanto, começou-se a observar inconsistências entre o que era pedido e o que era entregue.

Essas inconsistências entre o que foi pedido e o que estava na lista de tarefas foram criando situações complicadas. Foi perguntado ao Gerente de Projetos porque funcionalidades que haviam sido enviadas do começo do DS não estavam ali. Ele perguntava *"Quais?"* e a Especialista de Domínio não sabia dizer exatamente quais eram, pois ela trabalhava em requisitos apresentados em forma de estórias (chamadas no processo de épicos), que eram quebradas em pequenas tarefas. Ela "via" partes das estórias no projeto dadas como entregues e não "via" outras.

No entanto o foco era de preparação para a certificação e os esforços foram voltados para isso. As horas oferecidas pela consultoria haviam acabado e a consultora atendeu a empresa mesmo assim e ajudou muito na finalização da parte formal do modelo MPS.Br pois tinha muita experiência na aderência do MPS.Br ao SCRUM e esse era um ponto nevrálgico (liberdade *versus* rigidez de processos). Havia muito que se produzir em forma de textos e formulários e o Engenheiro de Software e Gerente de Projetos se dedicaram a isso.

Porém havia outras coisas a serem feitas que somente o Engenheiro de Software estava apto a fazer: Ele era o responsável por fazer as estórias e regras de negócio se transformarem em requisitos. Ele também atendia a demandas pontuais dos dois outros softwares da empresa. Na falta de alguém do setor

comercial, era ele quem fazia os treinamentos e, com a saída do Coordenador de DS, ele assumiu o acompanhamento da qualidade dos códigos escritos.

No processo de preparação para a certificação MPS.Br, a organização começa a usar a metodologia e vai registrando os pré-projetos, para verificar a evolução do modelo no atendimento às metas. Porém, para que a certificação se dê, os avaliadores conferem o atendimento às metas apenas dos três últimos projetos teste rodados.

Assim, antes de a empresa marcar oficialmente a vinda dos avaliadores, ela deve passar por uma acareação de três projetos (um finalizado, um em andamento e um que ainda vai entrar em produção) para ser liberada para agendar a visita oficial dos avaliadores. O objetivo é saber se a empresa está apta a pleitear a visita para certificação.

Era julho de 2015 e não foi possível rodar um dos pré-projetos de avaliação exigidos pela consultoria que acompanhava a implantação do MPS.Br. A data prevista para rodar e apresentar o projeto era na primeira quinzena de julho. Porém, a empresa não estava atendendo a todas as 44 metas de gerenciamento de requisitos e gerenciamento de projetos propostos pelo modelo. Nova força tarefa para atender essas exigências e agora sem a ajuda da Especialista de Domínio porque esta seria avaliadora interna.

Na metodologia de certificação na MPS.Br, a organização a ser avaliada recebe de 1 a 3 avaliadores que podem ficar até três dias fazendo a avaliação. Essa é feita a partir do seguimento de uma tabela de autoconfrontação que serve para verificar se existem evidências no processo de desenvolvimento da empresa que atendam às metas de maturidade. Além da avaliação externa da SOFTEX, a instituição deve, ainda, formar um avaliador interno e indicá-lo, formalmente, para acompanhar os avaliadores no processo.

O avaliador interno deve conhecer os processos de DS da empresa e deve ter mais de 70% de aproveitamento no curso C (aprovado pela SOFTEX). A

participação desse avaliador interno durante a avaliação é a de participar da avaliação e dirimir dúvidas pontuais do avaliador, caso haja.

Mesmo existindo esse momento adaptativo, o YesTeka já estava com seus módulos básicos rodando e em produção. Um cliente novo aceitou a proposta de testar o software, e um processo de automação foi iniciado para o cadastro dos livros dele. Um cliente antigo aceitou fazer os testes de migração. Um desenvolvedor procurou a gestão para dizer que tecnicamente o sistema estava rodando em uma base que não dava produtividade e ele estava preocupado. Porém, todos os módulos que os clientes estavam testando foram feitos dentro do modelo. Então, por que esses problemas?

A gestão procurou o Engenheiro de Software e reclamou da produtividade. Ele se posicionou, lembrando que estava sobrecarregado com todas as tarefas que ele deveria cumprir, mas justificou que, pelo que ele estava acompanhando, os processos estavam encadeados e que para a gestora saber o que estava acontecendo, bastava que abrisse o Redmine (plataforma onde as tarefas eram registradas por ele) e ver todas as apropriações de horas e tarefas que estavam sendo feitas.

Paralelo a isso, havia o outro sistema, o YesManutenção, que não era um software novo como o YesTeka, mas os pedidos de melhoria desse sistema também eram incluídos na pilha de tarefas já no formato MPS.Br. Para esse sistema havia um desenvolvedor e um auxiliar dedicados exclusivamente a ele e a empresa possuía, em seu quadro de funcionários, uma Engenheira para atender aos clientes do YesManutenção. A Engenheira se apropriou muito rapidamente do uso das ferramentas que o MPS.Br estava aplicado. Ela abria o Redmine, conferia se seus pedidos estavam na lista, cobrava prazos e verificava quando uma tarefa era dada como cumprida. Porém não conseguia avaliar se os códigos estavam em conformidade com os requisitos.

Em conversa com a gestão, a Engenheira também reclamava que os prazos não eram cumpridos e expressou que esperava que o modelo trouxesse comprometimento à equipe. Ela chegou em 2014 na empresa e se interessava,

especialmente, pelas melhorias nos processos de negócios e gerenciamento. E também estava muito alinhada com a consultora, com o Gerente de Projetos e com o Engenheiro de Software para a certificação do MPS.Br.

A gestora pediu ajuda à Engenheira para olharem juntas as tarefas descritas para o YesTeka e passou a fazer acompanhamentos mais ostensivamente de sua sala. *"Agora consigo "ver" que as coisas estão andando"* (Gestora). A Especialista de Domínio "olhava", mesmo porque ela participou da implantação dos modelos. Porém ela não "via" que se estava caminhando e reclamava com o Gerente de Projetos. Começou a haver um *"distanciamento entre os grupos porque eu não via as estórias que eu enviava em sua completude"* (Especialista de Domínio).

Assim, somente em agosto de 2015 a empresa de consultoria fez esse passo a passo com o setor de DS e definiu que a instituição estava apta a rodar os projetos que fariam parte dos três projetos a serem avaliados oficialmente. A esta altura, todos os processos estavam sendo documentados pelos Desenvolvedores e, mesmo havendo atraso no prazo inicial previsto para a certificação (por culpa da parada de dois meses para a construção de um módulo), a gestão e envolvidos estavam satisfeitos com o andamento.

A gestão mostrava contentamento, pois o modelo estava sendo "materializado". Porém, não parou de cobrar do Engenheiro de Software o cumprimento dos prazos. Em um dado momento, ele veio até a gestão para dizer que não estava satisfeito com a maneira como os Desenvolvedores estavam trabalhando e que havia problemas no DS. *"Eu pedi que me listasse o que o preocupava e, dos problemas relatados, os de relacionamento no setor, mais me preocuparam"* (Gestora).

Já o Gerente de Software estava mais preocupado com algumas questões "técnicas" e de gestão. Ele trouxe demandas de aquisição e implantação de ferramentas de teste automatizadas e também trouxe modelos de gestão de DS que poderiam ser implantados. Estava envolvido mais com os Desenvolvedores e com a execução das tarefas de cada projeto no tempo estimado. Apesar de o

Gerente de Projetos tentar junto à equipe fazer valer os prazos previstos, ele só via o projeto quando as tarefas já estavam quebradas, com estimativa prévia do Engenheiro de Software e já na plataforma Redmine para ser o esforço ser votado pela equipe.

A esta altura, a implantação do MPS.Br estava sendo *"tocada"* (Gestora) principalmente pelo Engenheiro de Software e pela Engenheira do responsável pelo YesManutenção. Por causa da data de avaliação que se aproximava, a Especialista de Domínio estava estudando mais a fundo as ferramentas que davam suporte à aplicação do MPS.Br e também estava mais próxima ao Gerente de Projetos para atender às dúvidas de escopo e validar tarefas.

Por estar, novamente, mais próxima aos processos internos, a Especialista de Domínio perguntou a ele sobre o porquê do não cumprimento dos prazos, já que a equipe tinha aumentado e estavam usando o modelo que eles mesmos pediram. Ele respondeu que a forma de organização interna não dava espaço para esse controle e que os Desenvolvedores tinham verbalizado isso para ele. *"Estou preocupado"* (Gerente de Projetos).

A Especialista de Domínio também recebeu essa informação com preocupação porque não parecia simples para ela entender que um processo daquela envergadura estava sendo obedecido em todas suas 44 metas e que isso não estava interferindo nas entregas finais das tarefas. O setor de DS já estava no processo de implantação do MPS.Br havia um ano e meio e já dominavam e estavam implantando a filosofia do modelo. Eles também participaram do processo de aderência do SCRUM ao MPS.Br.

Com essas questões sem respostas em mente, a Especialista de Domínio passou a participar *"com outro olhar"* das reuniões de alinhamento das metas do MPS.Br, que já estava em sua reta final. Em uma dessas reuniões o tema a ser tratado era o Cálculo de Estimativa de Tempo de Execução de Tarefas. Era uma reunião importante porque o controle e gerenciamento do tempo são partes relevantes dentro do modelo para o Gerenciamento do Produto. Além disso, a data para receber os avaliadores para a certificação estava para ser marcada.

Nessa reunião estavam presentes a Consultora, a Especialista de Domínio, a Engenheira e a equipe de DS.

Em um dado momento da reunião, a consultora estava fazendo uma fala sobre estimativa de tempo reforçando a metodologia quando um dos Desenvolvedores falou:*"Ok, podemos até dizer que é essa a metodologia correta, eu também acho que devia ser assim, mas não é"*. Uns riram, outros ficaram sérios. A consultora perguntou *"Como assim?"*.

O Engenheiro de Software tentou intervir, mas todos olhavam para o Desenvolvedor esperando a resposta. Todos olharam para o Desenvolvedor e ele, mesmo constrangido, explicou que quando uma estimativa de tempo era pedida a eles já vinha junto uma pré-estimativa calculada pelo Engenheiro de Software e que eles sempre validavam aquela pré-estimativa como correta. *"Eu parei de respirar"* (Especialista de Domínio).

Parecia que do início ao fim dessa cadeia existiam problemas *"e esse era bem grave"*(Consultora). Ficou claro que, na maioria das vezes os Desenvolvedores não validavam a estimativa pelo tempo que acreditavam, validavam pelo tempo que era pré-calculado pelo Engenheiro de Software e levado a eles. *"O tempo que estava ali era o tempo estimado pelo ES, eu não me sentia à vontade para questionar, os outros, não sei"* (Desenvolvedor).

A consultora disse que isso tinha de ser corrigido com urgência para os futuros projetos. E que isso faria com que os novos projetos rodassem no modelo *"dar certo"*. Ela pediu esse comprometimento da equipe para que todos os futuros *Sprints* (ciclos do projeto) fossem cadastrados na plataforma. Todos concordaram. Foi feita uma ata para essa reunião elencando as decisões contidas nela. Esta também fazia parte do modelo e foi anexada aos documentos do DS e às plataformas de acompanhamento do MPS.Br.

Esta reunião desencadeou em uma situação inusitada. A gestora pediu para participar das reuniões de estimativa. A equipe de DS achou "estranho" mas, *"todos ali já trabalharam juntos e fazendo estimativas juntos no passado, qual*

o problema de eu estar lá?" (Gestora). O modelo não cabia estimativas de pessoas que não faziam parte do DS. No entanto, a equipe sugeriu que ela votasse com eles, mas que seu voto não ia *"valer".*

Então, a gestora passou a frequentar reuniões pontuais de estimativa de esforço. Em uma delas um projeto foi proposto: A criação de uma ferramenta interna de catalogação para o Bibliotecário dentro do YesTeka. Isso já acontecia no sistema anterior, no SofTeka. No entanto a Especialista de Domínio havia visto um modelo de tabela proposto por um consórcio de especialistas com representantes de todo mundo e com sede na Holanda. O modelo da tabela era *"bem completo"* e ela sugeriu que fosse usada no YesTeka. Porém, a gestora considerou "*que novo sistema poderia oferecer mais ao usuário"*.

Foi então decidido que um novo projeto seria feito para atender a essa demanda. O SofTeka e a ferramenta oferecida pelo consórcio, só mostravam uma lista de ocorrências dos assuntos que se queria tratar biblioteconomicamente. Agora, era preciso abrir automaticamente essa lista de assuntos da tabela, em todos os seus níveis e quantas vezes ou onde a expressão de busca (assunto) aparecesse. "*o classificador teria uma noção ampla dos assuntos e poderia decidir qual o número da tabela ele usaria no seu material*" (Gestora).

Acontece que as tabelas tradicionais de classificação (Classificação Decimal de Dewey – CDD e a Classificação Decimal Universal - CDU) possuem mais de mil e quinhentas páginas com milhares de assuntos e sub-assuntos contidos nelas. Essa nova funcionalidade deveria permitir ao catalogador *"Ver"* seus níveis e sub níveis quantas vezes o assunto aparecesse. Uma tabela tradicional não permite isso por questões físicas. O mesmo assunto pode ser contemplado na página 100, na 234, na 435, e na página 800, por exemplo. Construir essas funcionalidades de forma digital era, então, o diferencial almejado.

Assim, com as novas funcionalidades solicitadas, o Engenheiro de Software fez os requisitos, as tarefas e também a pré-estimativa. A estimativa geral é uma etapa da metodologia MPS.Br onde os requisitos já analisados e quebrados em tarefas são discutidos em reunião do setor de DS. A equipe, com cartas

numéricas em mãos, se expressa ao votar o quanto de esforço (esforço é uma grandeza da ES), seria gasto para cumprir tal tarefa.

A empresa possui um Projeto de Desenvolvimento formalizado (uma das exigências do modelo MPS.Br) e nele está explicitado que essa prática de estimativa está aderente ao modelo. As cartas para voto possuem os números de esforço 0,5 a 8. (0,5, 3, 5 e 8). A mensuração do esforço a ser direcionado para o cumprimento de uma tarefa mede mais que o tempo para executá-la. É um valor onde estão contidos cálculos que podem interferir no DS como um todo. O tempo para executar as tarefas vem a partir dessa medida de esforço.

O voto da gestão por tarefa era feita a partir de sua experiência de 16 anos na gestão de desenvolvimento de softwares de gerenciamento. Então, para a funcionalidade da nova tabela de assuntos, os Desenvolvedores votaram em esforço 3. *"Eu pensei em oito, mas como todos colocaram três, achei que havia superestimado a tarefa, mas três eu achei que não daria"* (Gestora).

A gestora relata que ela votou em esforço 8. A equipe tinha se comprometido a votar na estimativa que eles acreditavam ser a correta e não somente validar a pré-estimativa. A pré-estimativa era 3. Todos disseram que o esforço para cumprimento dessa tarefa era pequeno, que *"era três mesmo"* (Desenvolvedor).

Como a gestora participava, mas seu voto não contava como válido, o número 3 de esforço votado pelos Desenvolvedores permaneceu. O tempo de execução seria de 120 horas de um desenvolvedor dedicado integralmente à tarefa.

No MPS.Br, uma vez votado o esforço, a equipe se compromete a cumprir aquela tarefa dentro do projeto e do prazo estipulado. O fato de a gestora participar dessa parte do processo pesava para eles, pois eles sabiam que ela estava ali para reforçar que eles deviam estimar os esforços o mais próximo possível da possibilidade de seu cumprimento.

Porém, ao invés de 120 horas para a construção desse módulo foram gastos três meses. Outras entregas foram comprometidas e quatro desenvolvedores tiveram

de ser envolvidos na tarefa. A gestora cobrou a entrega da funcionalidade com veemência.

> Talvez eles não tivessem uma noção exata, ou do que estavam validando ou do que os requisitos estavam propondo". "Eu queria ver como funcionava essa coisa de estimativa. A gente implantou o modelo também porque antes as entregas quase nunca eram feitas nos prazos, agora parece que piorou" (Gestora).

O mês para a avaliação foi marcado: novembro de 2015. O projeto que possuía a criação da tabela de assunto seria analisado pelos avaliadores para a certificação e trazia um caso de erro de estimativa. De acordo com a Consultora, isso não se configurava um problema desde que não fosse uma situação recorrente e que fosse devidamente discutido com as equipes e documentado nas plataformas de controle. A Gestora não ficou satisfeita com essa situação, pois esperava prazos mais exatos com o modelo rodando em sua totalidade.

A presença dos avaliadores para a certificação demanda um investimento financeiro. A empresa já não dispunha de recursos, pois ao longo do processo foram feitos muitos investimentos em pessoal, equipamentos, ferramentas, servidores, nuvem etc. e o sistema vendável não ficou pronto na mesma velocidade. Como as coisas já estavam próximas da finalização, não fazia sentido voltar atrás, mas durante toda a implantação do MPS.Br e escrita do software, cada vez que se tomava conhecimento de atrasos, a Diretoria Financeira se posicionava perante o grupo:*"vocês já pensaram na possibilidade de esse software não ser viável?"*(Diretora financeira).

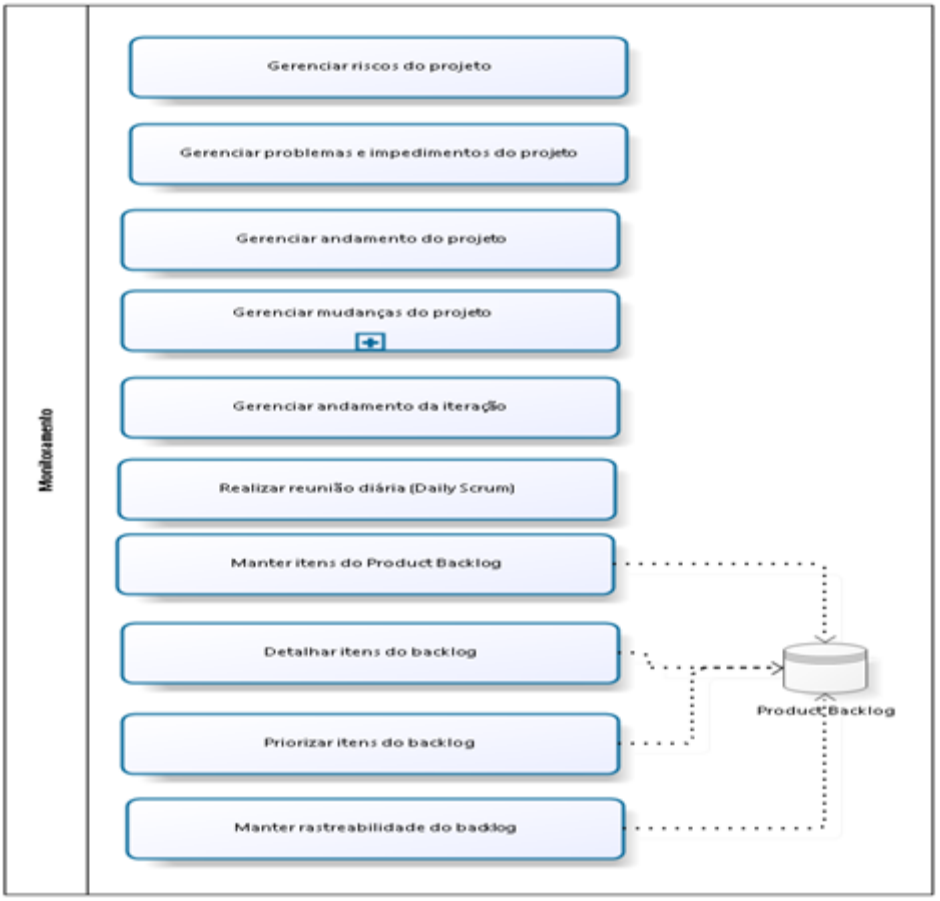
"Se as entregas não estão sendo feitas, então algo está errado no processo, pode ser a gestão pode ser a escrita dos códigos, pode ser no DS. O que não pode acontecer é que não tomemos uma atitude perante isso" (Diretora Financeira).

As inquietações tinham sua razão de ser por isso, já com a data de avaliação marcada a expectativa e a esperança foi que, com o uso sistemático do MPS.Br, *"a locomotiva andasse nos trilhos"* (Diretora Financeira), por isso era muito

importante para todos que tudo estivesse acontecendo da forma correta para a certificação.

O movimento de organização para o recebimento dos avaliadores estava seguindo em frente. As planilhas de confrontação para saber se as metas estavam sendo acolhidas e registradas estavam ficando prontas. Na plataforma de controle de tarefas já havia um grande número de estórias para que os requisitos fossem definidos e as tarefas listadas. Os Desenvolvedores continuavam a manter o processo de apropriação de horas e também faziam o registro de finalização de tarefas na plataforma de controle de registros, o Redmine.

FIGURA 7 - GUIA DE MONITORAMENTO DE PROJETOS DA EMPRESA



Fonte: Documentos cedidos pela empresa estudada

A esta altura, as funcionalidades que a Especialista de Domínio e a Gestora consideravam imprescindíveis para a construção do sistema vendável já faziam parte da pilha de priorizações. O processo já estava desenhado e podia ser monitorado por estas via sistema. FIG. 7.

Quando houve o primeiro conflito, a gestora designou uma desenvolvedora da equipe de DS para salvar todas as telas do SofTeka, legendá-las com palavras chave de regras de negócio e enviar para os Desenvolvedores.

Era setembro de 2015 e a certificação estava agendada para novembro. O Gerente de Projetos, em reunião com a gestão, informou que o apoio da desenvolvedora designada para fazer os testes estava sendo imprescindível. Ela estava ajudando na elucidação de dúvidas para a escrita dos códigos do YesTeka. A desenvolvedora em questão não codificava para o YesTeka por ser recém formada, porém, ela trabalhou nos serviços de informatização de acervos no sistema antigo e dominava as regras de negócio do software mais que todos.

A Especialista de Domínio havia se afastado do DS. Ela estava envolvida com a parte burocrática da certificação e, também, passou a viajar com a Gestora para vendas do sistema. A empresa *"não pode esperar mais por um retorno financeiro do YesTeka, temos de ir às vendas. Não temos mais pernas para esperar o idealizado"* (Diretora Financeira).

Devido à atratividade estética, pedagógica e de diferenciais interessantes para o meio biblioteconômico:

> Muitos clientes visitados compraram o sistema mesmo sabendo que algumas funcionalidades estavam em produção, mas a gente não estava confortável com aquilo. O prazo para a entrega do sistema com todas as funcionalidades básicas já havia expirado e sabíamos que, tão logo o sistema rodasse em larga escala, muitas funcionalidades que não estavam priorizadas deveriam ser contempladas" (Especialista de Domínio).

Essa parecia ser uma preocupação futura, mas na verdade, deveria ser uma preocupação imediata.

A Gestora e a Especialista de Domínio enviavam sistematicamente mais itens para abertura de projetos com novas funcionalidades ou com pedidos de melhorias das já existentes. Pensavam que com as pilhas de tarefas já listadas havia trabalho para a equipe se dedicar.

O Gerente de Projetos e o Engenheiro de Software pediam que as tarefas fossem priorizadas. Como elas não priorizaram as tarefas no sistema, houve um dia em que as duas estavam incomunicáveis por motivo de viagem e os Desenvolvedores ficaram parados por três dias. Isso aconteceu porque, de acordo com as normas do MPS.Br, os Desenvolvedores não podiam escolher tarefas que não estivessem priorizadas no sistema.

Quando a Gestora voltou de viagem e ficou sabendo do ocorrido, não viu com bons olhos aquela situação. Ela chamou o Gerente de Projetos e o Engenheiro de Software que lhe explicaram que não foi só por isso que pararam. *"Na verdade eles não estavam parados, eles estavam verificando erros e tentando corrigir".* (Engenheiro de Software). *"Uma luz vermelha se acendeu. Cinco desenvolvedores corrigindo códigos por três dias?"* (Especialista de Domínio). Ficou claro que, além da questão de prazos, a questão qualidade de códigos também deveria ser discutida *"mas com quem?"* (Gestora).

Até então ninguém da gestão havia pensado na qualidade do código. *"É esperado por toda empresa de DS que uma ou outra coisa seja revista, mas parar três dias eu nunca vi"* (Gestora). A empresa havia desenvolvido outros softwares sem seguir o modelo MPS.Br e a qualidade dos produtos eram muito bem recebidas pelos clientes demandantes. A Gestora pediu, então, ao Gerente de Projetos, que acompanhasse o Desenvolvimento dos Códigos mais de perto, porém essa não era a área dele. Tais situações gerenciais ocorridas dentro do setor de DS estavam sendo documentadas em formulários padronizados e de acordo com a proposta do modelo.

A gestora acompanhava pela plataforma de controle dos projetos, mas relatou que isso foi ficando extenuante dadas as outras funções que ela tinha dentro da empresa. Ela também citou o fato de as tarefas serem muito quebradas

dificultando que ela visse a feitura da funcionalidade no todo. Por causa disso, e contrariando a proposta do modelo, ela voltou a frequentar o setor de DS e perguntar o que cada um estava fazendo *"ficava mais fácil para eu juntar o que eles estavam fazendo pela fala"* (Gestora).

O Gerente de Projetos questionou a postura da Gestora, trazendo que com essa atitude ela poderia estar atrapalhando a linha de pensamento da equipe e, também, atrasando o projeto que tinha estimativa de esforço. A Gestora respondeu que era para ele contabilizar essa interrupção no *Buffer* (tempo deixado para situações imprevistas durante o desenvolvimento).

Ela via que o setor estava trabalhando muito, mas as entregas pareciam *"agulha num palheiro de tarefas a desenvolver"* (Gestora). Os desafios da empresa naquele contexto eram muitos: Econômicos, pois havia captado recurso em bancos duas vezes. Gerenciais, pois cada vez ficava mais claro que o setor de DS estava passando uma crise de confiabilidade na equipe, pois os prazos continuavam não sendo atendidos. De Relacionamento, pois devido aos prazos apertados a equipe não tinha muito tempo hábil para trocarem ideias e estudarem.

A Desenvolvedora que assumiu o acompanhamento da equipe no desenvolvimento do YesTeka, procurou a Gestora e a Especialista de domínio pedindo que fizessem as priorizações e ofereceu ajuda. As três chamaram o Gerente de Projetos e foram feitas as priorizações da pilha de tarefas que estavam no Redmine. A desenvolvedora passou então, a fazer testes nos códigos antes de serem considerados como "Entregues".

O Gerente de Projetos fazia relatórios mensais da atuação do setor de DS como um todo. Ali ele apresentava para os gestores as ações do setor. Em uma dessas apresentações foi observado que um dos itens avaliados por ele caiu muito: a produtividade. Questionado por essa repentina queda, ele considerou que foi por causa da Desenvolvedora que agora testava se as tarefas marcadas como prontas estavam mesmo funcionando como pedido no escopo. Se não atendiam, ela devolvia para o desenvolvedor consertar antes de passar para a próxima tarefa.

Na ocasião o Gerente de Projetos falou da necessidade de se ter ferramentas automatizadas de testes para acompanhar melhor o DS.

O que antes parecia inequívoco por estar elencado no modelo MPS.Br, começou a ser questionado. O setor de DS inteiro já havia passado três dias olhando códigos e parecia que implementar novas funcionalidades demorava cada vez mais. Começou-se a questionar o que "produtividade" queria dizer para cada um da cadeia de produção dentro da empresa. O que o setor de DS entendia por entregas e, principalmente, começou-se a questionar se, ao fazer a estimativa de esforço, a equipe de DS estava dando prazos para o desenvolvimento de códigos com qualidade.

Para a gestão e para os sócios da empresa era óbvio que a qualidade era o carro-chefe do processo. Mesmo porque estava sendo aplicado um modelo de Melhoria do Produto de Software. Mas, depois de tantos erros de estimativa do esforço e de tantos atrasos nas entregas por causa dos testes que detectavam erros de códigos que deviam ser corrigidos, *"muita coisa precisava ser reavaliada"* (Gestora).

A Gestora reportou tais preocupações ao Engenheiro de Software e ao Gerente de Projetos. Eles se reuniram com os Desenvolvedores que disseram que a qualidade estava sendo observada e que os problemas estavam na forma como os módulos tinham sido construídos e na plataforma de *framework* escolhido. *"No entanto, a mesma equipe que construiu o produto escolheu o framework e eu estou apenas tentando gerenciar o que já me foi dado pronto"* (Gerente de Projetos).

Em meio a tudo isso, a aplicação da MPS.Br seguia e, para que fosse bem-sucedida era necessário que a evolução dos projetos apresentados à empresa certificadora dessem resultados "concretos" e que o Gerenciamento de Requisitos e o Gerenciamento de Projetos estivessem sendo feitos da maneira correta. Esses resultados concretos se traduziam, de acordo com o modelo, em forma de resultados: estimativas corretas, baixo índice de refacção de códigos, códigos "limpos", satisfação do cliente final perante as entregas apresentadas,

pouco uso do tempo destinado a imprevistos, pouca incidência de horas extras, dentre outros.

Em relação aos requisitos para que houvesse qualidade o modelo esclarecia:

> A equipe de DS deve avaliar tecnicamente a viabilidade de cada requisito a ser implementado dado critérios objetivos:
> a) É implementável? Quando uma estimativa não é possível porque o requisito envolve alguma impossibilidade técnica (tecnologia ou conhecimento ausente do time), a equipe o rejeita.
> b) É testável? Cada item de projeto tem sua descrição que, juntamente com a conversa com o Engenheiro de Software, permite à equipe (atuando como desenvolvedor, tester ou documentador) testá-lo, dentro da previsão realizada.
> c) É rastreável? Todos os itens do projeto estão cadastrados Redmine com associações pertinentes corretas, permitindo sua consulta a rastreabilidade pela equipe, durante mudanças de projeto (para análise de impacto). Deste modo poderão ser facilmente rastreados com códigos fontes e também consultados para verificação de rastreabilidades diversas no Redmine, além de apontamento de trabalho (apropriação de horas).
> d) Está coerente com os demais requisitos do produto? Todas as tarefas do projeto são discutidas com a equipe e, caso haja inviabilidade na implementação de algum deles devido a inconsistências, esta é evidenciada na impossibilidade de estimativa. O fato do desenvolvimento se pautar em Iterações curtas (15 dias) facilita que o Engenheiro de Software segmente e os desenvolvedores estimem (e validem por consequência) blocos coerentes de requisitos.
> e) Está claro e completo? A descrição do requisito está de fácil entendimento para a equipe que, através da conversa complementar com o Engenheiro de Software consegue uma margem aceitável de previsão de tamanho em Pontos Empresa (PE). Caso seja necessário maior detalhamento do requisito, isto ainda pode ser feito durante outros momentos do modelo.
>
> Estas práticas em conjunto são suficientes para garantir que os requisitos estejam apropriados para o trabalho. Ou seja, garantem "por consequência" o atendimento a cada um dos critérios listados acima quando são apresentados à equipe durante a reunião.” (Plano de Desenvolvimento da Empresa, PE).

Assim, seguindo as premissas acima e já com o software em uso em algumas empresas vieram demandas de alguns clientes. Esses clientes eram parceiros antigos da empresa e aceitaram fazer a migração do sistema antigo para o YesTeka e reportar correções ou dúvidas de uso que tivessem. Eles traziam demandas diferenciadas, mas que tinham de ser atendidas de imediato. Por consequência, a priorização da pilha de tarefas teve de ser refeita e novas demandas foram acrescentadas. De acordo com a Gestora, a feitura dessas funcionalidades eram *“inegociáveis”*.

No modelo MPS.br, quando se muda o escopo de um projeto ou se acrescenta algo, o responsável pela gestão do DS deve avaliar se as mudanças cabem no tempo destinado a esse tipo de imprevisto (*Buffer*). Se não cabem, o esforço deve ser votado novamente. Durante o quarto, o quinto e, também, no último pré-projeto que estavam sendo rodados para testar o uso da metodologia e também para apresentação para a certificação, a Gestora e a Especialista de Domínio retiraram, mudaram as priorizações e pediram novas funcionalidades.

Como algumas funcionalidades foram tiradas, as novas funcionalidades pedidas talvez coubessem no tempo do *Buffer*. Os Desenvolvedores votaram o esforço para a implementação dessas tarefas e o Engenheiro de Software recalculou que as entregas seriam feitas no prazo para esses projetos. Porém ele fez isso contando com o uso completo do tempo do *Buffer* e também contando com o uso de horas extras dos funcionários todos os sábados. Mais horas extras, mais atestados, contudo os projetos foram finalizados nos prazos estimados.

Alguns Desenvolvedores foram, individualmente, até a sala da Gestora, e eles traziam a mesma demanda *"Precisamos melhorar os processos dentro do setor"*. O Gerente de Projetos também foi conversar individualmente com a gestora e levou uma análise que ele havia feito do setor como um todo. Ele disse que muitas coisas deviam ser tratadas e que algumas delas dependiam da gestão, como, por exemplo, se o modelo de produção do DS deveria continuar o mesmo e que ele mesmo tinha alguns modelos a apresentar.

A gestão, até então, jamais pensou em mudar o modelo que estava sendo implantado *"e seria preciso muita oratória para que eu pensasse em mudar"* (Gestora). Porém, era necessário entender o que estava acontecendo. Porque estavam falando sobre melhorar processos? Porque o Gerente de Projetos falou em mudar o modelo que nem havia sido certificado ainda? A gestora marcou uma reunião formal com toda equipe de Desenvolvedores, com o Engenheiro de Software, o Gerente de Processos e com a Especialista de Domínio. *"Era um quebra cabeças cheio de disse me disse. Ao final o cenário que eu via era de tensão, mas sem eles verbalizarem o que era"* (Gestora).

Mesmo assim, a aplicação do MPS.Br estava rodando em sua totalidade e as normas estavam sendo seguidas. Todavia, Já havia passado dois anos desde a proposta de fazer um novo sistema com previsão de entrega do produto *vendável* era para início de 2015. O ano estava finalizando e havia muito a ser feito ainda colocando a empresa em uma situação financeira complicada.

Faltava a construção dos módulos de relatórios, tipos de materiais similares a livros e materiais multimídia que já deveriam estar prontos, de acordo com as estimativas, há meses e para complicar a situação, muitos clientes antigos aguardavam a migração pelo fato do sistema rodar em ambiente web. Porém os Desenvolvedores ainda não tinham começado a codificar essas funcionalidades. Como migrar?

5.7 "De tarefa em tarefa não existia um criador"

Quando a empresa iniciou a implantação do MPS.Br, um dos interesses pelo modelo era a documentação e o controle do que se fazia no DS. Para atender a isso, a maioria das funcionalidades do YesTeka foi feita durante a implementação do MPS.Br. Dessa forma, os dados do ciclo de vida do projeto YesTeka estavam sendo devidamente registrados, documentados e com esforço estimado documentado.

A data da certificação estava se aproximando, as demandas dos clientes parceiros que estavam testando o sistema passaram a ser prioridade. Eram demandas variadas e houve mudança de escopo de projeto e recálculo do esforço. Além disso, era necessário iniciar as migrações dos clientes antigos. Havia muita preocupação se isso atrapalharia o processo de certificação e a Consultora foi contatada.

"Essas situações não são problema para a certificação, pois o modelo prevê imprevistos que vão além do estimado no Buffer desde que sejam discutidos, documentados e registrados" (Consultora). Então, em relação a essas mudanças no meio do processo, a empresa estava tranquila. Mas isso não impediu que as

cobranças gerenciais e da Especialista de Domínio fossem menos intensas. "*Tudo tem limites, precisamos do produto agora, e não esse faz refaz"* (Gestora).

O *feedback* dos clientes que estavam testando o sistema era muito bom. As funcionalidades pedidas por eles foram feitas, e as demandas externas para o software ficaram mínimas. *"Foi bom estabilizar, mas é inacreditável o que passamos aqui para chegar a isso. Agora é seguir em frente"* (Gestora).

O tamanho do software era grande porque a necessidade de integrar processos às novas formas de comunicação se fazia urgente no mercado. *"A parte "simples" do sistema já vinha bem completa e a parte "cereja do bolo" também já estava ficando pronta"* (Gestora). Agora era refinar e ampliar os tipos de materiais em 21 tipos que era o restante do escopo do projeto global e que já estava descrito na plataforma dentro dos padrões do MPS.Br.

O Engenheiro de Software e o Gerente de Projetos estavam dedicados a coisas diferentes. Um ao acompanhamento da alimentação dos dados necessários para os gerenciamentos listados no modelo MPS.Br, ao nível G de maturidade. O outro ao estudo de formas de melhorar a produtividade do setor desde que aderente a MPS.Br, pois a ideia de se mudar os processos não foi considerada. As novas tarefas estavam priorizadas e os relatórios mostravam as evoluções da equipe. Em meio a isso, um cliente considerado de grande porte pela empresa, entra em contato para dizer que eles só migrariam se o sistema tivesse a funcionalidade de Biometria para validação da circulação dos materiais nas bibliotecas deles.

Essa demanda parecia simples para esse cliente, pois o sistema antigo, o SofTeka, tinha a funcionalidade de biometria. Entretanto, essa funcionalidade não estava nas priorizações imediatas feitas para os próximos projetos que iam rodar e os Desenvolvedores, para piorar a situação, disseram que biometria WEB não era tão simples assim. Junto à nova demanda não prevista, veio uma oportunidade, também não prevista, com clientes de bibliotecas escolares da esfera pública e com uma rede de Gestores Prisionais Associados – GPA.

O sistema começava a *"se vender sozinho"* (Diretora Financeira) com suas facilidades para o usuário cadastrador e sua estética interativa. Já se somavam 12 clientes que estavam com o produto em uso e as propostas dos módulos com rede social estavam sendo muito bem aceitas. Começou-se a distribuir as senhas para que os clientes interessados testassem o sistema. Foram enviadas também, senhas para professores das escolas federais e estaduais de Biblioteconomia de Belo Horizonte, São Paulo, Curitiba, Rio de Janeiro e Salvador para testes mais aprofundados.

"Era preciso saber se nossas escolhas foram certas" (Especialista de Domínio). Foram quebrados paradigmas da biblioteconomia, inserida uma nova lógica computacional para reuso da informação em sistemas de bibliotecas e colocadas tabelas interativas para entendimento do novo código de catalogação de materiais utilizados. Comprou-se um serviço do *Google* para fornecimento de imagens de capas de livros evitando que o catalogador saísse do sistema para buscar e inserir capas no registro do material e isso incidiu muito no trabalho do catalogador e indexador de livros.

As universidades deram devolutivas muito positivas e a empresa foi convidada a apresentar o sistema aos estudantes da Escola de Ciência da Informação da UFMG, da UNESP e para a equipe do Mestrado Profissional da UDESC em Florianópolis. Os serviços inseridos foram reconhecidos como muito úteis e os novos clientes estavam cadastrando em tempo muito menor que o usual para o mesmo processo no software anterior.

Quando a empresa apresentava os diferenciais do sistema para novos clientes, ela também citava que estava participando do processo de certificação para o nível G de maturidade do modelo MPS.Br. Essa informação era relevante porque na maioria das vezes que o sistema era apresentado a um cliente potencial um representante da TI participava da reunião. Esses representantes sempre elogiavam a iniciativa. Assim, até então a gestão estava satisfeita com a decisão pela implantação do MPS.Br, pois reverberava em outros setores dos

clientes visitados. Porém, a Diretoria Financeira sinalizou que os recursos destinados ao projeto já haviam acabado. Era hora de certificar.

Os três projetos selecionados para a certificação estavam engajados nos ciclos de vida pedidos pela certificadora (um projeto finalizado, um em produção e um iniciando o planejamento), e os registros na planilha de confrontação para os avaliadores da empresa certificadora estavam passando por revisão de escrita. Durante tal processo, a Especialista de Domínio teve uma dúvida sobre como os requisitos de uma funcionalidade específica foram descritos pelo Engenheiro de Software e onde eles estavam no processo como um todo. O Engenheiro de Software e o Gerente de Processos não conseguiram fazer a rastreabilidade do requisito no projeto de imediato. Houve um conflito ali, pois a empresa estava focada há meses na finalização da implantação e rastreabilidade era um aspecto importante para a organização.

"Fica difícil saber se não podemos confiar no processo ou não podemos confiar nas pessoas. Quando vocês iam nos contar?" (Especialista de Domínio). Isso gerou um desconforto porque foi extrapolada a barreira hierárquica e o tom da Especialista de Domínio foi agressivo. A Especialista de Domínio pediu à Gestora para participar das reuniões previstas no MPS.Br entre Gerente de Projetos e gestão. O pedido foi acatado.

A Gestora acatou a ideia porque queria *"alinhar os discursos*". A Especialista de Domínio queria saber se o que acontecia internamente no DS estava sendo relatado à gestão, por causa do caso da rastreabilidade. O Engenheiro de Software viu ali uma oportunidade para que a Especialista de Domínio ficasse mais próxima à gestão. *"Eu achei bom, pois a especialista sabe o que estamos produzindo e entende nossas dificuldades quando coisas são mudadas no meio do processo"* (Engenheiro de Software).

Todo esse movimento foi feito porque a rastreabilidade era um valor para a empresa, e também porque gerenciar os códigos fazia parte das prerrogativas do MPS.Br. Isso era reforçado todo o tempo pela Gestora, pela Especialista de Domínio e pela Consultora. *"Eu achava que entendia o que se passava no DS,*

mas não. Nas reuniões com a gestão, entram tantas situações, tantos conflitos, tantas demandas que eu não consigo ver o espelho do setor" (Especialista de Domínio). De qualquer forma, e apesar dos interesses diversos, a decisão ajudou no fluxo de informações da evolução da implantação do modelo uma vez que o Gerente de Projetos passou a informair melhor a gestora acerca de problemas cotidianos que estavam influenciando o DS e a aplicação do MPS.Br.

Se ter conhecimento sobre a que projeto uma tarefa pertencia e a quais outros ela estava vinculada era uma prerrogativa importante, porque não estava acontecendo de forma satisfatória? Mais uma reunião com a Consultora. Foi relatada para ela essa sensação de que o modelo não estava sendo cumprido. A Consultora abriu o sistema e fez uma pesquisa de requisitos. Em parceria com o Engenheiro de Software e Consultora, foi pensado um modelo de rastreabilidade usando *hashtags*. O modelo de rastreabilidade passou a ser aplicado aos projetos.

A Consultora informou que para a avaliação no nível G de maturidade aquela proposta atendia bem, mas que havia muitas ferramentas interessantes que poderiam ser usadas. Ficou acordado com o Gerente de Projetos que ele faria essa pesquisa juntamente com uma pesquisa para uso de ferramentas de testes automatizados. Os testes feitos pela Desenvolvedora eram manuais. Eles ainda estavam sendo feitos e, aliados aos imprevistos que estavam sendo documentados, continuavam "*atrasando as entregas"* (Gerente de Projetos).

"Uma coisa são os desafios que temos de enfrentar no dia a dia, outra coisa é a implantação do MPS.Br" (Gestora). Realmente parecia que as coisas caminhavam em separado. Os Desenvolvedores continuavam produzindo de forma a garantir a aplicação do modelo, mas o "dia a dia" deles não parecia ser simples assim.

Já haviam acontecido reuniões com todos os Desenvolvedores e a gestora conseguiu montar um *pool* de olhares diferentes sobre o mesmo processo. O Gerente de Projetos estava angustiado, temia que Desenvolvedores bons procurassem novo emprego porque estavam *"Num ritmo de trabalho muito*

exaustivo" (Gerente de Projetos). Junto a isso, a equipe começava a dar novos sinais de que os relacionamentos internos não iam bem.

Sempre que algo novo era pensado ou construído por um desenvolvedor específico, esse era muito exaltado pelas Sócias que gostavam de ver ideias novas serem aplicadas nos produtos da empresa. Havia muito tempo que isso não acontecia, pois de tarefa em tarefa não existia um criador. Existia uma tarefa com tempo médio estipulado de execução. *"Sinto falta da vibração no DS"* (Engenheira de Software).

Havia comentários paralelos e a equipe que estava desde o começo do projeto do YesTeka mudou o comportamento. Faziam as tarefas e seguiam o protocolo, mas conversavam menos entre si no horário de trabalho. Os outros dois, que entraram para fazer com que o sistema fosse desenvolvido mais rápido, *"Produziam, mas com muitas brincadeiras e ironias"* (Gerente de Projetos).

Finalmente faltavam três dias para a avaliação e certificação no MPS.Br. Novas reuniões com o setor de DS, novamente as questões de prazos e, nesse momento, os Desenvolvedores pediram uma reunião com o Gerente de Projetos para dizer que eles não estavam satisfeitos com a forma como os requisitos chegavam para eles. Disseram que faltavam pedaços, que estavam mal escritos, que eles nem sempre entendiam o que se estava sendo pedido e que eles "*Não podiam garantir a qualidade do produto assim".*

O Engenheiro de Software levou a situação à gestão que questionou a metodologia do Gerente de Projetos para gerenciar a equipe. *"Se eles tem de seguir protocolos, regras, monitoramentos e se eles se comprometem com isso no processo, onde está seu acompanhamento para fazê-los entender que o momento de questionar os requisitos é no início de cada projeto, e não no final de seis projetos completos entregues?".*

O Gerente de Projetos não esclareceu seu papel gerencial para a situação e a gestão disse que não entendia as inconsistências relatadas pelo próprio DS. *"Olhando os projetos em execução no Redmine eu não vejo nenhum problema*

no Software". (Engenheira de Manutenção). Essa parte foi dita em uma reunião. E foi feita a pergunta: *"Quem consegue ver problemas nos códigos olhando o projeto em execução?"*.*"Eu, mas a equipe é muito boa e tem autonomia para fazer, pesquisar, pensar"* (Gerente de Software).

Sobre fazer, pesquisar, criar: "*Eu não vejo eles fazendo isso. Não com empolgação"* (Gestora).*"Talvez seja porque antes eles faziam uma previsão de tempo para a parte do projeto que eles iam pegar. Mas eles mesmos dizem que a metodologia é correta porque a empresa não fica refém de um desenvolvedor apenas"* (Engenheira de Manutenção).

5.8 A certificação

A visitação para a certificação se deu de forma muito organizada. Testes, verificações, e-mails para orientação de datas e horários, confirmações e apoio da consultoria aconteceram de forma muito satisfatória. A empresa já havia rodado todos os pré-testes para a avaliação e todos estavam confiantes.

Para esse momento, vieram além da consultora que acompanhou o processo, também o diretor geral da FUMSOFT (empresa que forneceu os serviços de consultoria para a certificação) e todos foram unânimes que a empresa estava em conformidade com o modelo MPS.Br e apta para receber os consultores da SOFTEX para que avaliassem se a empresa estava apta a receber a Certificação do Nível G de Maturidade.

Assim, a avaliação aconteceu conforme o previsto. Foram analisadas todas as 44 metas de gerenciamento de projetos e gerenciamento de requisitos e a planilha de confrontação foi seguida e debatida em sua totalidade. A Especialista de Domínio foi avaliadora interna no processo. Foram feitas entrevistas com os Desenvolvedores, Gerente de Projetos, Engenheiro de Software e Gestora. "*Eles demonstraram muito conhecimento do processo e do jargão e sabiam da importância da equipe para o andamento dos projetos solicitados"* (Avaliadora SOFTEX).

Então, em novembro de 2015, a empresa passou a ser uma empresa certificada, nível G, no modelo MPS.Br e com recomendações para iniciar o processo de aumento de nível de maturidade, saltando o nível F, que já estava praticamente contemplado na avaliação feita, e pleitear o Nível E.

5.9 O que aconteceu depois

Os Desenvolvedores e demais participantes do processo de certificação sempre apoiaram a aplicação do modelo MPS.Br. No entanto, agora que ele era uma realidade e imediatamente depois da certificação, toda a equipe do DS ficou quase quinze dias sem seguir nenhuma regra do modelo MPS.Br e nem do SCRUM.

Porém, a desenvolvedora que fazia os testes precisava que eles *"apropriassem das horas para liberar testes para mim"*. E ela mesma chamou a atenção da equipe para isso e pediu que voltassem a seguir as orientações do Plano de DS da empresa. Agora era o momento de manutenção do modelo e eles recomeçaram com as reuniões, registros das horas e documentações.

Nos sete meses seguintes, o DS continuou seguindo as metodologias. Porém, os mesmos problemas citados ao longo do processo de implantação se mostravam desafiadores: cobranças da gestão e dos sócios referentes a prazos, muitas inclusões de funcionalidades na pilha de priorizações, mudanças de priorizações e cobrança da desenvolvedora que estava fazendo os testes para que os Desenvolvedores não entregassem tarefas incompletas.

Em meio a isso, um novo módulo devia começar a ser desenvolvido: o módulo para Periódicos. Além desse novo desafio, uma grande rede iria migrar o sistema e precisaria de parâmetros de moderação de ações dentro do software mais rígidos que os implementados até o momento. A Especialista de Domínio que agora estava mais dedicada às vendas do produto e ao desenvolvimento dessa pesquisa para a Qualificação deste trabalho, junto à banca, voltou a frequentar mais o setor de DS.

Por esse motivo, e em relação à manutenção do modelo MPS.Br ela percebeu que os Desenvolvedores continuavam a produzir os códigos para as tarefas designadas e continuavam os casos de entregas que eram feitas além do prazo estimado. Mesmo assim, e contrário ao que se via acontecer, a tendência era de se acreditar que a estimativa seria cumprida no próximo projeto, uma vez que estavam cumprindo os protocolos do modelo.

Infelizmente, o que aconteceu no primeiro semestre de 2016, (mais de dois anos após o início do desenvolvimento de um software que tinha previsão de entrega de seus módulos principais em 2015), gerou *"a maior crise que a empresa jamais pensou viver"* (Gestora).

A equipe comunicou à gestão que o software (que já estava rodando em várias empresas), tinha de ser refeito em outra plataforma de *framework* mais flexível. *"Que tinha de se começar do zero"* (Desenvolvedor).

6 O RECONHECIMENTO DO FRACASSO E A BUSCA POR SOLUÇÕES

Entendemos como importante contextualizar o leitor sobre a fase crítica da investigação e mote que acabou por interferir nos rumos desta pesquisa. Esse capítulo começa pelo que será o final deste relato: de quando a empresa foi comunicada que o software deveria ser refeito em outra plataforma e de como isso culminou no que foi chamado pela equipe de **"A Sala de Guerra"**.

6.1 Como tudo começou

O que deflagrou que a equipe verbalizasse o que estava acontecendo foi uma demanda da pesquisadora para o setor de DS. No segundo semestre de 2015, a pesquisadora havia feito um trabalho para uma disciplina do Mestrado sobre *Análise do Curso da Ação das Atividades de Trabalho* e o fez na própria empresa. Os módulos básicos do software para cadastro de livros já estavam prontos, e duas empresas parceiras já o estavam testando. Além dessas duas parceiras a empresa também terceirizou a informatização de uma terceira biblioteca e os livros estavam sendo processados tecnicamente na sede da organização. Essa situação facilitou a escolha por se fazer a análise do Curso da Ação do Cadastro de Livros e Cadastro de Autores pela pesquisadora.

À época, o resultado desse estudo foi que algumas das funcionalidades poderiam ser mais intuitivas e que a forma como o Cadastro de Autores estava apresentado induzia o cadastrador, mesmo experiente, ao erro. O Cadastro de Autores no sistema, em especial, também levava o cadastrador a um cansaço devido aos diversos cliques que esse deveria fazer para fazer a entrada de dados de livros com autores múltiplos (mais de três autores).

Para verificar a viabilidade de se implementar as propostas do documento, o trabalho acadêmico e seus resultados foram mostrados aos Desenvolvedores e à

Gestora. Como a conclusão da análise trazia cinco recomendações para enriquecimento da usabilidade do software (habilitar o Enter em quatro campos para confirmar as ações de entrada de dados e marcar de vermelho o autor principal), o foco foi em saber se eles concordavam que as modificações deveriam ser feitas.

Assim, Desenvolvedores e Gestora se dividiram em posicionamentos tais como: "*O cliente também tem de se adaptar, essa é a forma mais moderna de entrada de dados que existe"* e em frases como "*Sim, podemos fazer", ou mesmo: "Coloque a lista na pilha e priorize para dezembro"* (Gestora). E ao final das discussões a gestão determinou que as modificações fossem feitas.

A lista com cinco itens, a princípio, pareceu muito simples e rápida de ser atendida. A feitura desses artefatos foi considerada muito importante pela Gestora e Especialista de Domínio, pois prometia um impacto positivo na entrada de dados em massa e na qualidade de vida dos cadastradores. Além disso, pelo fato de a empresa prestar os serviços de informatização em bibliotecas, essas melhorias dariam melhor qualidade e produtividade ao resultado dos cadastros de livros e acervos.

Como a gestão concordou em atender às sugestões da pesquisadora, as tarefas de melhoria foram colocadas na pilha e também foram rediscutidas com os Desenvolvedores as motivações de se acatar as mudanças. A Desenvolvedora que fazia os testes no YesTeka foi uma entusiasta das mudanças no sistema. Isto por que ela fez treinamentos no software e considerou as mudanças sugeridas como de grande ajuda para os processos de entrada de dados de títulos e autores no sistema. A data para se fazer essas melhorias era dezembro de 2015.

Porém, dezembro acabou e chegamos a janeiro do ano seguinte. A Especialista de Domínio cobrou as mudanças ao Gerente de Projetos. Ele disse que as alterações estavam na fila, e que tão logo as tarefas da Rede Social fossem entregues, eles iniciariam "os estudos" para fazer essas modificações. *"Quando ele falou em iniciar estudos, soou muito estranho. As tarefas eram colocar um*

Enter em quatro campos e marcar de vermelho o autor principal" (Especialista de Domínio)

Essa estranheza passou despercebida. E só foi compreendida por todos bem mais tarde. Em fevereiro de 2016, a pesquisadora passou a fazer parte do grupo societário da empresa. Mesmo isso não sendo um impedimento para as relações entre ela e o setor de DS, estava claro, mesmo que de maneira implícita, que as hierarquias haviam mudado.

Um exemplo disso, é que as funcionalidades pedidas ainda não haviam sido entregues e a recém-chegada sócia ficou constrangida *"Em ficar cobrando todo dia"*. Mesmo por que, essa nova posição na empresa também era peculiar para a sócia e ela não queria fazer aquilo parecer uma "ordem". Esse foi um dos momentos em que as três *personas* (pesquisadora, sócia, funcionária), se confundiram.

Nesse ínterim, os processos rodando dentro da MPS.Br já tinham certa consistência e os ciclos de vida dos projetos estavam acontecendo. Também os atrasos. *"Nunca foi verbalizado em entrevista, por nenhum Desenvolvedor, que os atrasos ocorriam por causa do modelo"* (Pesquisadora). Mas, agora, não parecia tão interessante e produtivo ver em painéis e gráficos que o setor de DS não estava atendendo ao propósito do processo. Nem para eles, nem para a gestão.

Como foi dito antes, os objetivos em se apoiar a certificação eram diversos. De maior controle e documentação de um lado e do comprometimento da gestão em validar e manter as escolhas do outro. A essa altura gestão e setor de DS já haviam percebido que as coisas não eram bem assim. Porém, *"Se o modelo estava sendo seguido, porque as priorizações das funcionalidades para a continuidade da pesquisa da análise do trabalho ainda não estavam prontas?"* (Especialista de Domínio).

A essa altura e passado o momento de certificação, o Engenheiro de Software seguiu dando continuidade ao acompanhamento da equipe na escrita de tarefas

no Redmine e no acompanhamento da documentação do modelo. A pilha de funcionalidades a serem inseridas estava bem completa e as dúvidas diretas sobre o sistema eram passadas para a Desenvolvedora que estava fazendo os testes manuais no sistema e que agora passou a fazer a ponte entre os setores.

Foi quando a pesquisadora, ao se matricular em outra disciplina do Mestrado, decidiu dar continuidade ao estudo feito em 2015 e analisar se as mudanças feitas durante o exercício de Análise do Curso da Ação, na disciplina anterior, tiveram um efeito positivo para a empresa. Para isso deveria ter as funcionalidades prontas a tempo de fazer o estudo do impacto das implementações e redigir o trabalho.

A pesquisadora chama o Gerente de Projetos e explica a ele que precisa que sejam feitas as mudanças para atender a uma proposta acadêmica e ele pediu o prazo de abril. Porém, ao final do mês de abril as funcionalidades não tinham sido iniciadas. Maio e junho também não. A justificativa era que os Desenvolvedores estavam estabilizando algumas situações de erro de sistema que apareceram e estavam estudando como implantar as funcionalidades pedidas por ela.

Então, a pesquisadora foi ao Engenheiro de Software e pediu a ele para colocar no Redmine prioridade máxima para essas tarefas. O Engenheiro de Software procurou o Gerente de Projetos e ele respondeu que fazer o que estava sendo pedido não era tão simples como se pensava. E disse também que o desenvolvimento fez o sistema em uma base que não aceitava o Enter como confirmação de uma ação. *"Porque isso não foi dito antes? Eu poderia fazer o trabalho em outro sistema"* (Pesquisadora).

O Gerente de Projetos reportou à gestora tais dificuldades, porém a Gestora e a Desenvolvedora responsável pelos testes queriam as melhorias. Ambas acompanhavam automações há anos e sabiam que era necessário cercar o software com métodos de usabilidade que evitassem cansaço e, principalmente, evitasse erros. A Gestora não abriu mão da feitura das implementações. A partir desse apoio da gestão, a pesquisadora pediu uma estimativa e o Gerente de

Projetos colocou a priorização e a estimativa solicitando uma semana para implementar as cinco melhorias.

À revelia de todos esses acontecimentos, o prazo de entrega das funcionalidades prontas e testadas passou sem que fossem feitas. O prazo de entrega do trabalho para a disciplina do Mestrado também estava chegando e a Pesquisadora acabou por entregar ao Professor um relatório narrando esse fato e perguntou a ele se podia complementar o trabalho tão logo as melhorias fossem feitas e testadas.

O Professor aceitou uma segunda entrega. Esta procurou a gestão e relatou que, mesmo no Redmine e com priorização, as melhorias não estavam prontas e que ela precisava de uma estimativa *"real"* de quanto tempo demoraria, pois havia se comprometido e fazer uma segunda entrega do trabalho. Era julho de 2016.

Por esse motivo, a Gestora chamou o Gerente de Projetos e a Especialista de Domínio (e Pesquisadora e sócia) para conversarem. A Pesquisadora, disse ao Gerente de Projetos que ela não estava entendendo porque não foi feito. A Gestora perguntou à pesquisadora por que ela tinha deixado a coisa chegar a esse ponto sem reportar as não entregas a ela. Foi um momento complicado, porque a Pesquisadora acreditava ser uma situação pontual e pessoal. A gestora perguntou ao Gerente de Projetos: "*Quanto tempo para fazer?"*, ele respondeu que cinco dias, porém não com um, mas com todos os Desenvolvedores dedicados, um a cada combo.

Como tudo parecia muito atípico, a Especialista de Domínio perguntou ao Gerente de Projetos se havia algum problema pessoal, que ele estava muito distante e que seis meses era muito tempo para não se conseguir entregar *"tão pouca coisa"*. Ele disse que absolutamente não e que ele estava preocupado por que os Desenvolvedores estavam demorando muito mais que o esperado para realizar tarefas simples. Não apenas aquelas, mas outras também. Ele disse, ainda que ele iria conversar com a equipe sobre isso e que também iria conversar com a Gestora e levar a situação até ela.

A Gestora falou ao Gerente de Projetos que eles podiam parar a semana para se dedicarem a isso. Então, as tarefas ligadas ao modelo MPS.Br foram suspensas temporariamente e as práticas internas foram redimensionadas.Cinco Desenvolvedores estavam dedicados às tarefas da lista. Ao final de uma semana apenas uma estava pronta.

Reportar esse fato à gestora foi muito complicado para o Gerente de Projetos. Afinal de contas ele não a procurou apenas para dizer que as funcionalidades não estavam prontas. Ele foi até ela, em nome de toda equipe do DS, dizer que o sistema estava rodando sem erros ou problemas. Porém, da forma como ele estava construído não comportava nenhuma mudança. Ele disse à Gestora que na opinião dele o YesTeka estava inoperável.

6.2 A Sala de Guerra

Assim, em julho de 2016 a empresa interrompeu a produção do software. O que pareceria impossível, devido ao seguimento das premissas de gerenciamento de produto e gerenciamento de projetos, GRE e GPR do modelo MPS.Br aconteceu: o sistema estava atrasado, inoperável e muito acima dos custos previstos.

Após quase três anos de estudos e trabalho para construção do produto, divididos entre pesquisas nacionais e internacionais, escrita de requisitos, conquista do certificado e uso da metodologia MPS.Br, a equipe de DS procurou a gestão e verbalizou que a construção de novas funcionalidades estava cada vez mais complexa e que não conseguiam implementar nada em tempo minimamente aceitável ou razoável.

Era uma situação muito delicada e, para gerenciar esse momento de crise, a gestora pediu ao Desenvolvedor que havia sinalizado em um dos conflitos que havia problemas no DS, que ele se juntasse ao Gerente de Projetos e demais Desenvolvedores da equipe para que eles fizessem uma proposta de ação para a situação.

O Desenvolvedor em questão já havia mostrado para a gestão, anteriormente, as vantagens de se programar usando a linguagem *Phyton* como ferramenta no lugar do PHP utilizado para o desenvolvimento do YesTeka. Como os outros Desenvolvedores também tinham opinião sobre a melhor plataforma, foi pedido que fosse feita uma apresentação englobando outras possibilidades. Eles partilhariam ideias entre o grupo, produziriam e apresentariam essas propostas juntos.

No entanto, um elo importante das relações interpessoais havia se quebrado: o da confiança. Além do Engenheiro de Software que acompanhou todo o processo confiando na qualidade dos códigos criados, ninguém mais da gestão e diretorias era, ou Desenvolvedor ou Analista de Sistemas ou tinha qualquer outra competência que pudesse avaliar, com segurança, o que seria apresentado pelos Desenvolvedores.

Uma das sócias verbalizou essa sensação de incômodo perante avaliar as propostas da equipe e perguntou se não era prudente chamar alguém que entendesse de DS mais profundamente. A Gestora disse que pensaria, e, na sequência, ela ligou para a Ex-Consultora da implantação do MPS.Br relatando o que aconteceu e perguntando se ela podia participar dessa apresentação.

A Consultora disse que poderia vir, mas que também não tinha conhecimento suficiente de DS no que se referia a linguagens novas, *Framewoks*, plataformas e codificação para fazer essa avaliação. Por isso, ela sugeriu e contatou uma pessoa que ela sabia ter o perfil necessário. A pessoa indicada por ela aceitou participar desse momento. A Consultora se prontificou a ir também.

Ninguém da equipe do DS foi informado dessas presenças. *"Eu queria que eles escrevessem sem o compromisso de ser para um colega desenvolvedor. Só assim eu poderia entender um pouco de como essa equipe funciona hoje, eles não eram os mesmos do começo do processo"* (Gestora).

Por se tratar de uma situação de urgência, foi acordado com a equipe três dias para o estudo e preparação do material. Eles já tinham muita coisa, pois já

tinham apresentado *frameworks* entre si nos horários para formação livre do setor. A empresa tem um auditório completo com 36 lugares, onde eles faziam formações internas e também convidavam profissionais dos grupos externos deles para fazer palestras, encontros ou formações.

A comunicação da inoperabilidade aconteceu em uma segunda-feira, a apresentação dos modelos de solução foi agendada para a sexta-feira da mesma semana. Somente na sexta-feira, a equipe de Desenvolvimento de Software foi informada de que viriam consultores convidados para a apresentação. Eles preparam uma sala maior para receber o grupo.

O consultor convidado é um Desenvolvedor que trabalha em uma empresa de médio porte e é coordenador da equipe de DS dessa empresa. Possui Mestrado em DS, coordena 26 Desenvolvedores e também faz palestras e consultorias pelo país. Ele ouviu a narrativa dos sócios, desde o início da decisão por fazer o software usando o MPS.Br até o momento em que os Desenvolvedores declararam que não conseguiam fazer implementações básicas devido a forma como o banco de dados do software foi estruturado. Ele disse que teria que ver o que poderia estar acontecendo porque eram problemas inaceitáveis nessa dimensão se eles estivessem usado corretamente o modelo MPS.Br ao longo do processo.

Algumas coisas relativas ao código, a gestão tinha ciência de que deveriam ser refatoradas, pois o Engenheiro de Software e o Gerente de Projetos já haviam sinalizado. Dois Desenvolveres também tinham sinalizado isso quando buscaram a gestão em outro momento. Porém, *"ninguém falou de problemas a ponto de impedir a evolução do sistema*" (Gestora). Ao ouvir os sócios, o Consultor também perguntou sobre os valores investidos, onde os recursos foram captados, qual a situação financeira atual da empresa e quanto de fôlego financeiro ainda se tinha para o projeto.

As informações pedidas foram passadas ao Consultor: investimento na certificação, contratações para o projeto, juros com captação de recursos, dívida ativa em bancos e salários. Também foram passadas as planilhas de

investimento na compra de servidores, contratação de serviços da *Amazon*, investimento em protocolos de segurança da informação e em ferramentas adquiridas para atender às peculiaridades do sistema, tais como a parceria com o Google para comprar imagens de capas de livros de forma automatizada.

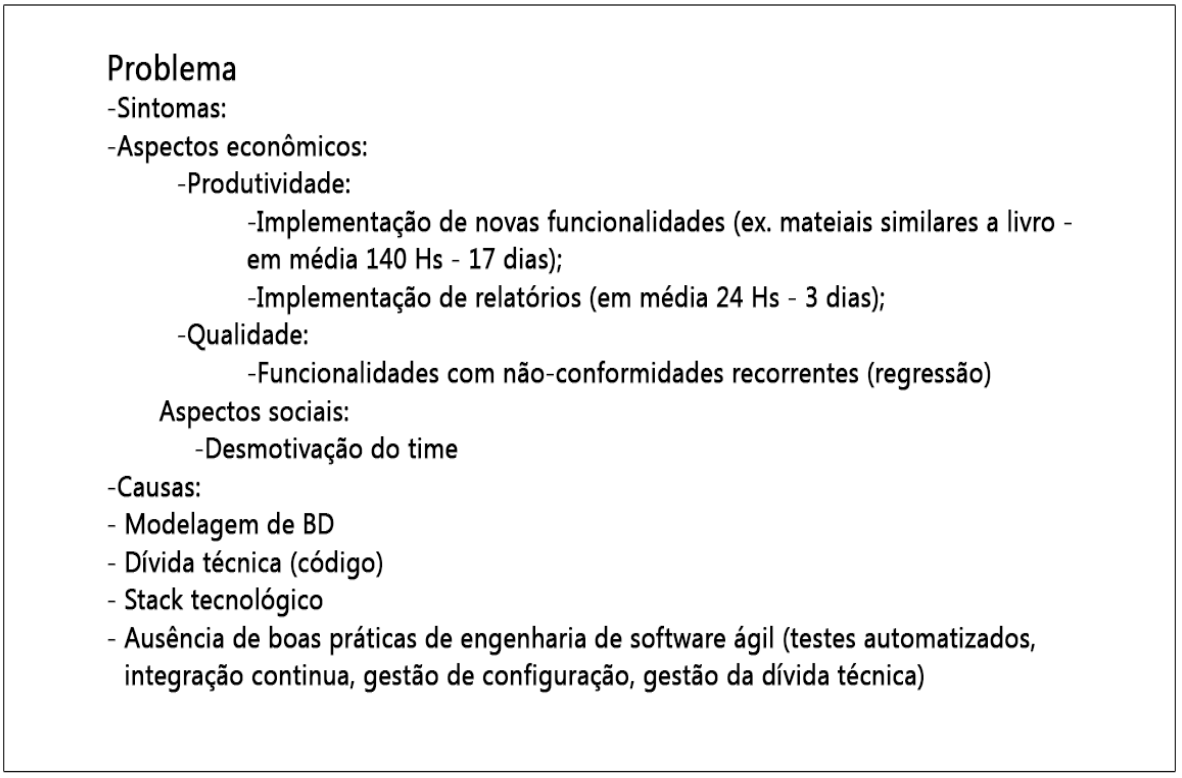
A reunião começou às 14h da sexta-feira. Com essas informações em mãos o Consultor foi apresentado à Equipe de DS. A Consultora já era conhecida pela equipe e além da equipe do DS a Gestora, a Engenheira e a Especialista de Domínio participaram da reunião.

Quando o consultor foi apresentado o grupo ficou tenso *"Eu me senti, pessoalmente, avaliado"* (Desenvolvedor), porém o Gerente de Projetos conhecia o Consultor, o que *"quebrou um pouco o gelo"* (ex-Consultora MPS.Br). A Gestora abriu a reunião falando sobre sua última conversa com a equipe de DS e do impacto que isso causou na empresa como um todo. *"Quando me falaram em novo sistema, começando do zero eu me desestabilizei. Agora que a ficha caiu eu precisava ter calma e entender quais os próximos passos possíveis porque eu tinha de dar retorno dessa situação para a Diretoria Financeira" (Gestora).*

Os Desenvolvedores estavam bem preparados para a apresentação. Levaram informações, gráficos, tabelas. Explicaram as opções atuais de mercado. Reforçaram que o *framework* que estava sendo usado, o *Zend*, havia sido descontinuado no ano anterior e que eles já vinham sinalizando para o Engenheiro de Software, Gerente de Projetos e Gestora que a plataforma deveria ser mudada para melhorar a produtividade.

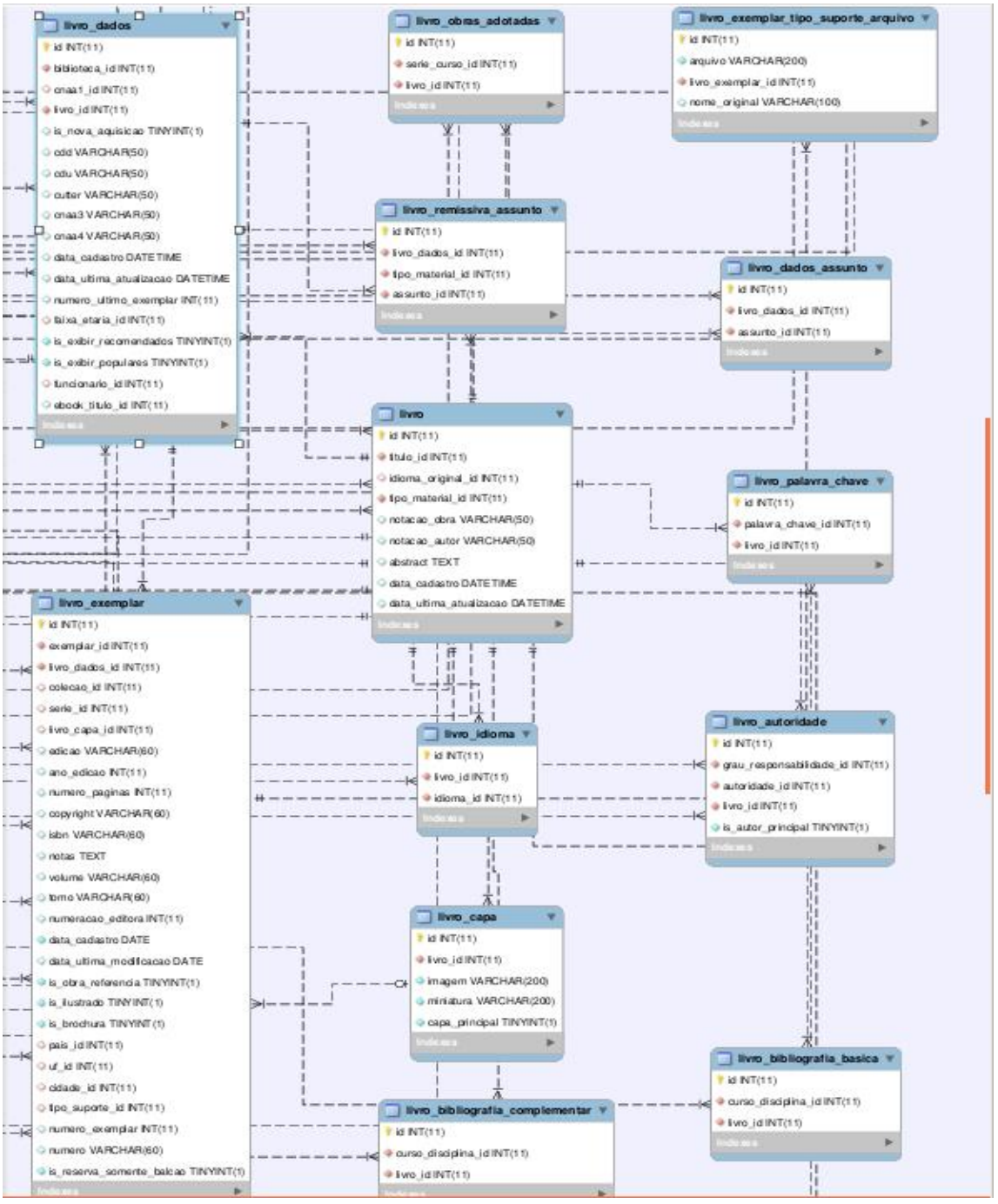
Porém, chamou a atenção da Gestora e da Especialista de Domínio que os dados como foram apresentados naquele momento, jamais haviam sido mostrados ou discutidos antes. Também causou espécie a todos que esses dados vinham desde o início dos registros nas plataformas de controle, mas não *"estavam nelas"*. Abaixo reproduziremos, alguns dos slides apresentados na reunião:

FIGURA 8 - JUSTIFICATIVA PARA REMODELAGEM



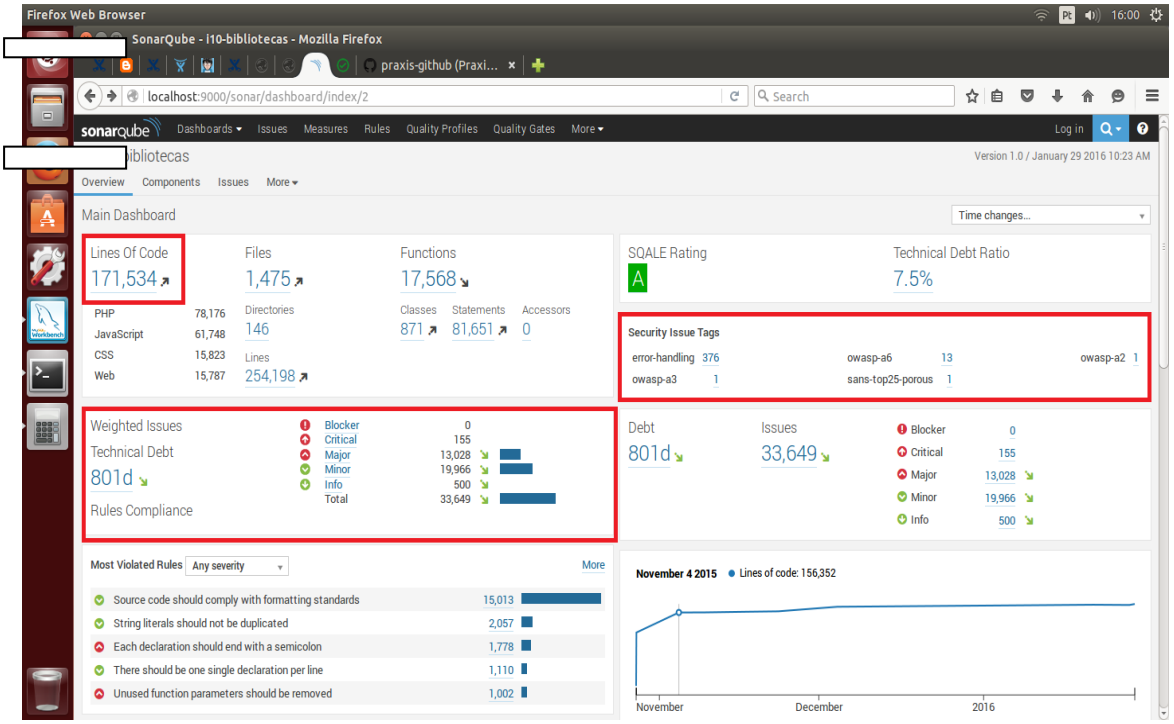
Fonte: Slides apresentados na reunião com o DS.

FIGURA 9 - MODELAGEM DO BANCO DE DADOS EM 2016



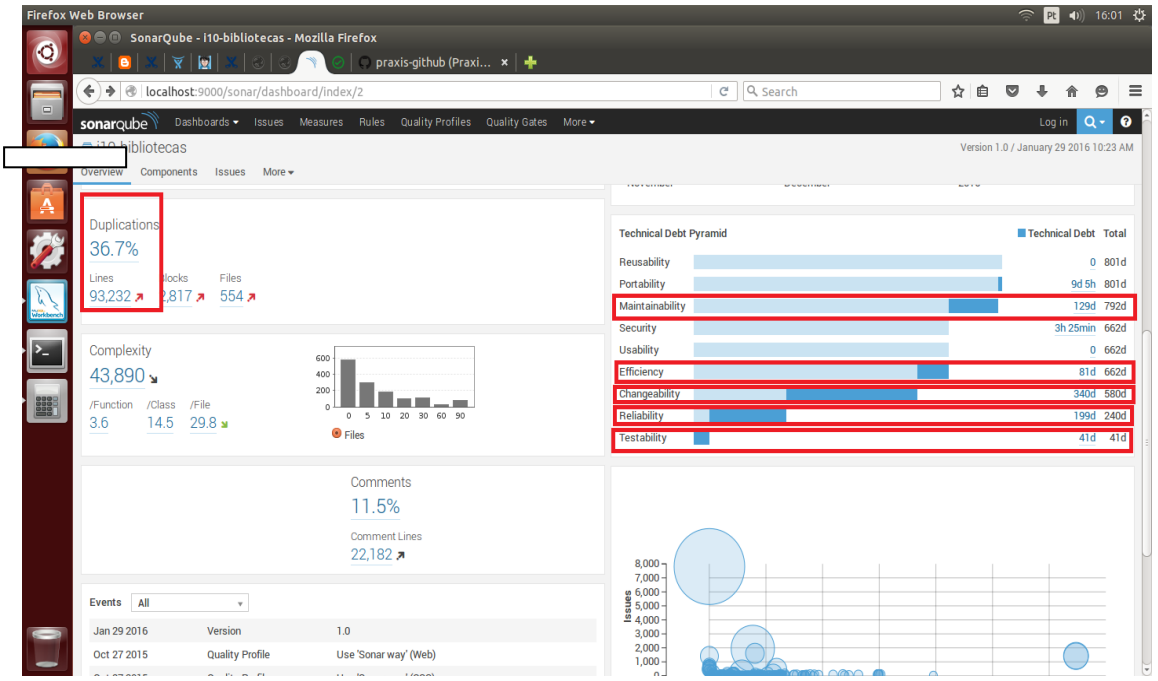
Fonte: Slides apresentados na reunião com o DS

FIGURA 10 - GRÁFICO DE DÍVIDA TÉCNICA



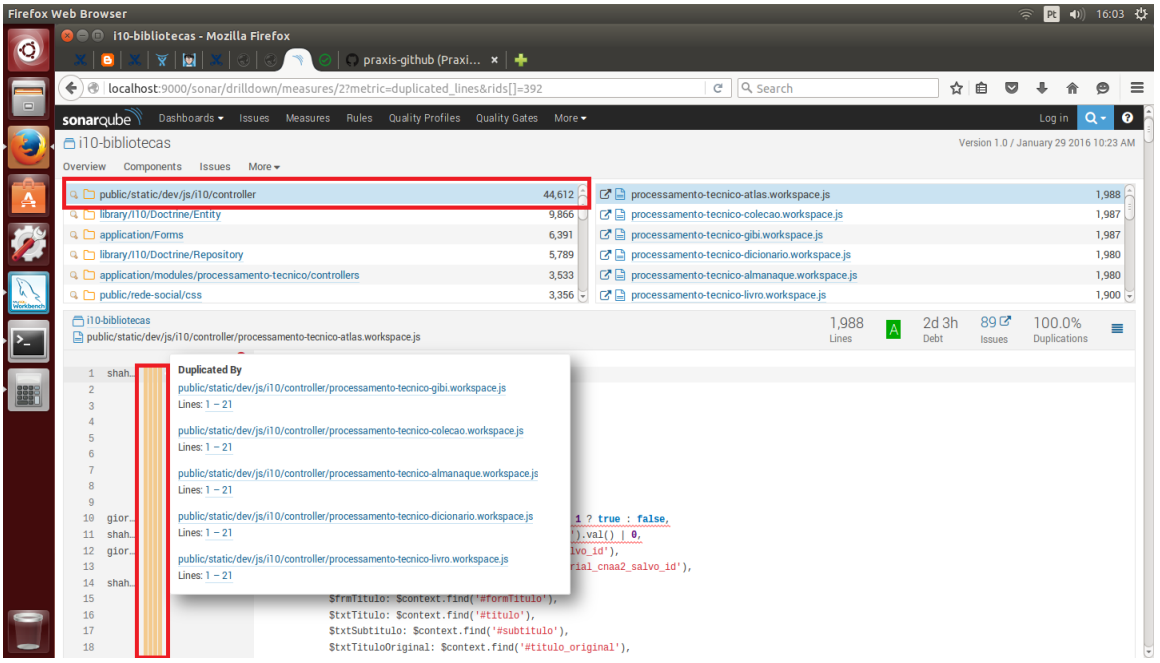
Fonte: Slides apresentados na reunião com o DS

FIGURA 11 - GRÁFICOS DUPLICAÇÕES DE LINHAS DE CÓDIGO



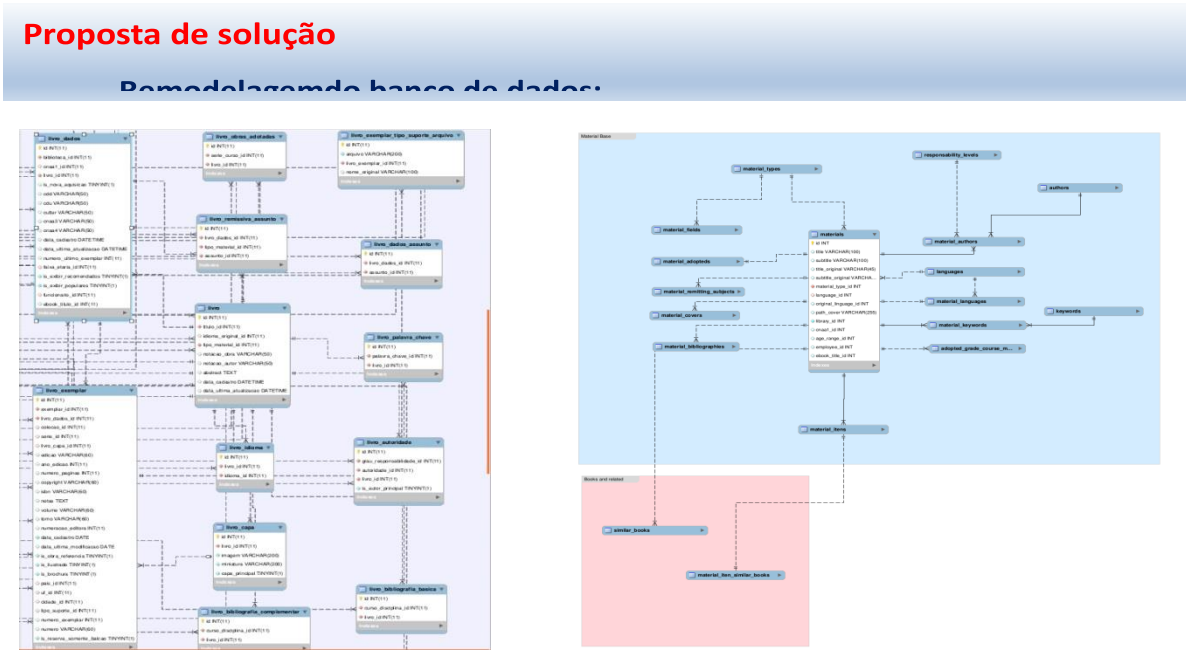
Fonte: Fonte: Slides apresentados na reunião com o DS.

FIGURA 12 - DÍVIDA TÉCNICA



Fonte: Slides apresentados na reunião com o DS

FIGURA 13 - MODELAGEM PROPOSTA



Fonte: Slides apresentados na reunião com o DS.

Um dos itens sobre a causa dos problemas citados pela equipe de DS foi muito contraditório aos ouvidos dos gestores. No documento apresentado estava

explicitado, que uma das causas era a “AUSÊNCIA DE BOAS PRÁTICAS DE ENGENHARIA DE SOFTWARE”, FIG. 8.. Bem, eles haviam pedido um modelo, escolheram qual queriam (o MPS.Br), implementaram, estavam utilizando e consideravam ali, a partir de um registro oficial, escrito, validado e apresentado pela equipe que havia “ausência de boas práticas de ES”.

Acrescido a isso, os gráficos mostravam que um número de linhas de código para as funcionalidades estava exagerado e fora do padrão usual. Sobre isso, o Engenheiro de Software fez uma fala, em tom de desabafo:

> Eu jamais achei que havia problemas com a qualidade dos códigos, o tempo todo confiei que eles entendiam que todo o movimento, tempo e dinheiro investidos no MPS.Br era para melhoria do produto como um todo. Principalmente o dos códigos”. “A sensação que tive naquele momento era que apenas o Gerente de Projetos, além dos desenvolvedores, tinha informação suficiente para entender o tamanho do problema (Especialista de Domínio).

Nesse documento, também foi apresentada uma tabela com as defesas de cada tipo de *framework FIG* 14 e 15. Foram enfatizadas vantagens e desvantagens do uso de três *framework*s selecionados por eles. Porém, o que chamou mesmo a atenção do grupo não foi o fato de ser importante mudar o *framework*, mas, sim, que em todas as alternativas de sugestões eles abarcaram a remodelagem completa da base de dados, analisando somente qual *framework* usar para a empreitada de remodelagem.

FIGURA 14 - OPÇÕES SUGERIDAS PARA SOLUCIONAR O PROBLEMA

Comparativo de alternativas			
Contra:	Reestruturação BD+ Zend	Reestruturação BD+ Laravel	Reestruturação BD+ Django
	- Baixa performance - Framework pouco produtivo; - O código fonte não será refatorado;	- Não é tão produtivo quanto o Django; - Curva de aprendizado da equipe e apenas 1 pessoa (Lucca - Risco) possui conhecimento da ferramenta; - Apesar de já ter uma comunidade grande e ativa, não é tão fácil achar um desenvolvedor PHP que possua conhecimentos desse framework. A mão-de-obra é mais cara. - A engine de template do framework, o Blade, é bem diferente do Twig. O Twig, que é a engine de template que utilizamos hoje é inspirada na engine de template do Django. Então para migrar do Twig para o Blade tende a ser mais complexo; - Período inicial para reestruturação da base de desenvolvimento (Configurar vagrant, criar models, adaptar template e estrutura de cruds, configurar db multitenance) / capacitação - Não prevê suporte automatizado para testes (unidade e integrado)	- Curva de aprendizado da equipe e apenas 1 pessoa (Naison - Risco) possui conhecimento da ferramenta. - Mão de obra em média 20% mais cara do que PHP, mas é fácil a capacitação para quem conhece o PHP. - Período inicial para reestruturação da base de desenvolvimento (Configurar vagrant, criar models, adaptar template e estrutura de cruds, configurar db multitenance) / Capacitação

Fonte: Slides apresentados na reunião com o DS.

FIGURA 15 - COMPARATIVO DE SUGESTÕES PARA SOLUCIONAR O PROBLEMA

Alternativas de soluções			
Pros:	Reestruturação BD+ Zend	Reestruturação BD+ Laravel	Reestruturação BD+ Django
	- Conhecimento do time; - Infraestrutura atual já está toda integrada.	O framework possui ORM próprio, nativo, o Eloquent. O Eloquent é um ORM bastante performático e bem ligado ao framework - Possui muitas validações prontas, com o texto de erro facilmente editável. O framework contém uma engine de template chamada Blade. O grande objetivo é reduzir a quantidade de código PHP inserido no meio do HTML e consequentemente aumentar o reuso - Documentação completa e rica em detalhes.	- Linguagem de desenvolvimento ágil (recursos comum utilizados no dia a dia já são implementadas nativamente ou possui algo já pronto). Ex: Envio de arquivos para aws, e-mails e etc. - Facilidade na implementação de infra/deploy. - Curva de aprendizagem baixa - Ferramentas integradas com AWS. - Documentação completa e rica em detalhes - Performance - Desenvolvimento orientado a testes - Seguro (Implementa os principais problemas de segurança) - Usando por empresa: Globo.com, Instagram, Pinterest - Será realizado testes automatizados (unidade e integração)

Fonte: Slides apresentados na reunião com o DS

Estes slides apresentados para o grupo traziam a explicação de como e porque eles pediam mudança de Banco de Dados e *Framework*, os prós e os contras, nesse momento, eles já trouxeram causas do problema e soluções, mas não mencionaram como as coisas chegaram àquele ponto. Como conclusão a equipe sinalizou que a melhor solução seria a opção de reestruturação do banco de dados usando a linguagem *Phyton* unido ao *framework Django*. Na apresentação a equipe de DS falou sobre datas. A equipe disse que para fazer a transição com a estruturação do banco de dados tanto no *Zend*, quanto *Laravel*, quanto *Django,* o tempo gasto seria praticamente o mesmo (estimativa dada pela equipe de 5 a 6 meses para a reestruturação + *framework*).

Porém eles não colocaram essa estimativa nos slides da apresentação, apenas verbalizaram e registraram a conclusão nos slides. Então, de acordo com eles, a opção pela reestruturação do banco de dados e utilização da linguagem *Phyton* e *framework Django,* demoraria o mesmo tempo para ser implantada que se fosse feita a refatoração das funcionalidades com problemas, utilizando o *Zend* ou o *Laravel*. Tudo, de acordo com a equipe de DS, aconteceria em torno de seis meses.

Diante de tal cenário, foram várias as reações dos diversos atores. A Especialista de Domínio, seguindo as premissas e tempos expostos no quadro preparado pelo setor de DS, achou que a opção pela linguagem mais atual (*Phyton*) era a melhor. A Gestora ponderou que ela não poderia somente pensar nos tempos de execução colocados ali por eles, que ela tinha de pensar também na qualidade do novo produto e nos custos da decisão pela mudança. *"As coisas tem de ser consideradas no seu todo, eles parecem não ter ideia do quanto custa a uma empresa pequena parar por seis meses"* (Consultora). *"Eu estava muito decepcionado com a equipe, achei que a qualidade da escrita dos códigos não podia chegar a esse ponto. E não acho que o problema seja apenas o Zend"* (Engenheiro de Software).

O Gerente de Projetos concordava com a opção dada pela equipe de DS de reiniciar a escrita do sistema em outra linguagem. O Consultor, que até então não havia se pronunciado, ouviu as explanações e fez algumas perguntas muito

pontuais. As respostas foram dadas, *"mas nem todas satisfatórias"* (Desenvolvedora). O Consultor pediu informações mais precisas, pediu desdobramentos sobre o número de linhas que cada tarefa tinha e disse que voltaria no sábado para "*ver*" o código.

Os sentimentos eram variados ao término da reunião. *"Os desenvolvedores pareciam animados com a possibilidade de usar nova plataforma para programação"* (Engenheira). *"Eu esperei o posicionamento do Consultor, mas já tinha uma opinião"* (Gestora). O Consultor falou à Gestora que queria conversar com Gerente de Projetos e com o Engenheiro de Software.

O Gerente de Projetos apoiava as propostas de se mudar totalmente de *framework*. O consultor fez uma reunião com a Gestora e demais sócios. Nessa conversa, ele colocou os pontos de vista dele sobre a situação. Ele disse a todos que o *Phyton* tinha boas promessas como linguagem, mas que achava que o que levou àquela situação era mais que isso. Discorreu sobre a importância dos requisitos, das regras, do envolvimento das equipes entre si e do compromisso entre eles e empresa.

O Consultor também chamou a atenção dos Sócios para a seguinte ponderação:

> Se para usar qualquer dos três *framework*s o tempo vai ser o mesmo, então todo o quadro montado por eles foi feito para desviar atenção que o verdadeiro problema ali não era o f*ramework* e sim, como eles planejaram e executaram o banco de dados que agora eles querem remodelar.

O Consultor ponderou também nessa conversa, que o momento com o Engenheiro de Software e com o Gerente de Projetos serviu para ele entender um pouco o processo. Outro ponto forte desse encontro, de acordo com o Consultor, foi que a conversa elucidou dúvidas sobre o questionamento que a gestão havia feito pelo fato de o uso do modelo MPS.Br não ter *"impedido esse desfecho"*.

Este disse que achava que as estimativas mostradas não estavam *"realistas"* e que nenhum dos Desenvolvedores soube dizer como chegaram a elas. Ele

lembrou à gestão que um dos pontos fracos citados para o uso do *Django + Phyton* demonstrado pela equipe de DS, era que essa dobradinha de ferramentas ainda era de domínio de poucos profissionais. A Gestora lembrou que, de acordo com o quadro apresentado pela equipe, um programador com domínio de *Phyton* custava para a empresa aproximadamente 20% a mais no mercado. O Consultor concordou que é assim e trouxe que apenas um dos Desenvolvedores da empresa sabia programar nessa linguagem.

Isso queria dizer que se a opção pela remodelagem fosse aceita, além de se voltar ao início do processo, a empresa teria de treinar seus Desenvolvedores e que essa curva de aprendizagem pode levar tempo. *"Para coroar a avalanche de informações que estavam chegando, o que me vinha em mente o tempo todo era: quem vai avaliar a qualidade do produto?"* (Gestora).

O Gerente de Projetos estava empenhado junto com a equipe em resolver a situação da volta da produtividade. A ausência do Engenheiro de Software impactava no suporte aos outros sistemas da empresa. Os Desenvolvedores trabalharam em melhorias requisitadas para os outros softwares enquanto esperavam um direcionamento.

Era julho de 2016 e qualquer posicionamento que a gestão pensava para solucionar o desafio esbarrava nos recursos financeiros e na confiança se o que foi estimado realmente ocorreria. Era possível vender o sistema, mas era preciso saber quanto tempo levaria para darem continuidade às funcionalidades do planejamento inicial.

O Consultor acompanhou esse movimento. Na quarta-feira, quatro dias úteis depois dessas reuniões, ele foi à empresa e a Gestora disse a ele que precisava que as ações de DS continuassem e que a evolução do software seguisse dentro do que os Desenvolvedores já tinham conhecimento. E que nessa perspectiva, ela não estava vendo um melhor caminho. A Gestora também conversou com o Gerente de Projetos. Ela não tinha viabilidade econômica para parar tudo e todos seis meses e também não acreditava nos prazos estimados.

Assim, o Consultor e Gerente de Projetos conversaram com a equipe, e, "depois de muitas ponderações verificações e trocas" foi sugerida a refatoração do sistema migrando do *Zend* para o *framework Symfony* (que até então não tinha entrado nas discussões), mas que todos conheciam. O *Symfony* é um dos *framework*s mais populares hoje, no Brasil.

O Consultor avaliou as informações dadas pelos Desenvolvedores na reunião e também em entrevistas específicas e mensurou juntamente com o Gerente de Projetos que essa refatoração duraria 30 dias com toda equipe do Yesteka dedicada à resolução dos problemas que estavam dificultando a inclusão de novas funcionalidades no sistema. Depois desse período, essa ação seria continuada concomitantemente à feitura das tarefas listadas no Redmine, até a que a modelagem estivesse de acordo com os padrões de usabilidade, flexibilidade e manutenibilidade explicitados no Plano de Desenvolvimento da Empresa.

Com essa nova proposta em mãos, a gestora foi comunicada dessa alternativa e acatou a proposta. Em entrevista sobre esse momento: *"Vamos fazer a refatoração como vocês a repensaram. Mas eu gostaria de saber como vocês vão fazer isso, pois não temos fôlego para bancar financeiramente alguma coisa mal dimensionada"* (Gestora). A essa pergunta, eles responderam que para isso também tinham uma proposta.

É então que, nesse momento, nasce a ***Sala de Guerra***. Uma metodologia de forma de trabalho temporária, pesquisada e sugerida pelo Consultor, equipe de Desenvolvedores e Gerente de Projetos como uma tentativa de retomar as ações de DS no tempo estimado.

Para dar vazão a essa proposta, a equipe passou a trabalhar em um espaço físico único para refazer, juntos, partes do código. O objetivo desse momento radical, de acordo com o Consultor e com o Gerente de Projetos, era o de adequar o código às possibilidades de mudanças de projeto ao longo do tempo, sem impactos que inviabilizassem a continuidade do sistema.

Durante os dias seguintes, a equipe estudou e discutiu o código entre si. Verificaram possibilidades e validavam em conjunto as decisões. O nome Sala de Guerra vem de uma terminologia da Administração e pretende fazer com que ações pontuais sejam resolvidas em um período curto e limitado de tempo.

"Eu precisava que eles ficassem focados, mas ao mesmo tempo eles tinham de ter paradas rápidas e descontraídas várias vezes ao longo do dia" (Gerente de Projetos). Por isso eles instalaram na sala um tipo de cronômetro que ajudava para que houvesse essas paradas. O restante da empresa (Diretoria, RH, Comercial, Suporte) via de fora esse *"cronômetro rodando"* (Suporte), com muita estranheza, pois, a empresa como um todo era bem flexível com horários e com a circulação de todos em todos os setores ao longo do dia.

O Desenvolvedor que sugeriu o *Phyton* não ficou muito convencido que essa era a melhor opção. Ele não acreditava muito no sucesso da Sala de Guerra. Porém ele foi um dos maiores colaboradores para o sucesso da empreitada e um dos mais ativos dos desenvolveres ao *"se envolver no processo e dar resultados"* (Gerente de Projetos).

Passados os 30 dias o sistema estava operável e a equipe voltou a seus postos de origem. Estavam cansados, mas viam um sistema e o resto da empresa também. O Engenheiro de Software assumiu o atendimento ao sistema YesManutenção, pois a equipe não seguiu as regras do modelo MPR.Br para fazer os reparos no sistema. Os Desenvolvedores continuavam usando o Redmine, e eles mesmos ou a Desenvolvedora que fazia os testes começaram a alimentar as pilhas e as tarefas (agora maiores), na plataforma.

O Gerente de Projetos, no acompanhamento da execução das tarefas retomou as reuniões diárias com a gestora e, em agosto, a lista pedida pela Especialista de Domínio foi desenvolvida e entregue. As novas funcionalidades foram testadas e colocadas em produção.

A mesma pessoa que participou das entrevistas de autoconfrontação para os módulos de entrada de título e autor no sistema, em 2015, fez o teste de

usabilidade das novas funcionalidades *"da tão fadada lista"* (Pesquisadora). Ela avaliou as mudanças sugeridas pelo Trabalho de Análise do Curso da Ação, como muito positivas e produtivas. Mas para todos ficou a incerteza sobre o que realmente aconteceu para que tantas adversidades ocorressem durante um processo que teve a implementação, de certa forma, considerada tranquila.

6.3 Dilemas da "maturidade"

A consequência da **Sala de Guerra** e seus desdobramentos foram um grande imprevisto para toda a empresa. Até mesmo para a Pesquisadora, que esperava fazer um estudo basal (e bem-sucedido) de implantação de um modelo de melhoria de software em paralelo com a criação e desenvolvimento de um sistema de gerenciamento informatizado.

Causou espécie à Pesquisadora esse momento, considerado por toda empresa, como muito crítico, uma vez que a maioria dos entrevistados afirmaram que sabiam que partes do código não estavam de acordo com os níveis de qualidade da empresa e sabiam que isso comprometia, não só o projeto como um todo, mas, também, a saúde financeira da instituição.

De uma certa forma, parecia haver uma unanimidade em reconhecer que a aplicação do MPS.Br não garantia, necessariamente que os resultados seriam à altura do projeto. Isso se percebeu quando disseram que não havia boas práticas de desenvolvimento de software no documento de pedido remodelagem do software. Para alguns, depois da vivência de toda essa situação, o modelo parecia *"trazer consigo um ideal inalcançável e tóxico"* (Especialista de Domínio). Mas, ao mesmo tempo, ninguém, mesmo tendo acesso ao longo do processo a bibliografias e estudos críticos sobre DS, conseguia pensar uma proposta que não fosse rígida e burocrática para controle e acompanhamento de processos.

Na verdade, tantas ponderações positivas e negativas foram feitas após esse processo, que simplesmente interromper o uso da metodologia parecia para a

gestão e para as equipes uma espécie de *"volta à infância"* (Gestora). *"Não pode parar, gastou-se muito dinheiro, olha a imagem para mercado*" (Diretora Financeira). O fato de a empresa estar sendo pesquisada no quesito Desenvolvimento também interferiu nessas ponderações, pois as reflexões acerca de processos internos de DS saíram dos muros do setor de Desenvolvimento para todos os espaços da empresa.

Os dilemas para continuidade do modelo como é formatado hoje, continuam desafiando as tomadas de decisão da empresa. Mas numa tentativa de melhorar a maneira como os processos estavam sendo vivenciados e, tentando preservar a natureza mercadológica que o modelo MPS.Br traz, em outubro de 2016, a Gestora informou ao setor de DS que eles deveriam se preparar para retomar a aplicação do modelo MPS.Br. Só que, dessa vez, com *"mais maturidade por parte das pessoas, e não do modelo*".

No capítulo seguinte, faremos a discussão e conclusão dos dados acima relatados a partir de nossa percepção sobre os impactos que a aplicação do MPS.Br trouxe às formas de trabalho dentro da empresa. Faremos essa explanação tendo como pano de fundo as mudanças nas lógicas de produção, relações, comunicações e resultados percebidos no percurso do desenvolvimento do novo software e como essas mudanças acabaram por interferir no resultado final da produção do software.

7 DISCUSSÃO E CONCLUSÃO

O Processo de Desenvolvimento de um Software esbarra em momentos de indagação e reflexão acerca das condições que limitam, inibem ou facilitam a comunicação entre as Comunidades envolvidas. Tradicionalmente durante esse processo, os requisitos de sistema desejados são comunicados aos desenvolvedores de modo a permitir a criação de produtos alinhados com as especificações e visão, geralmente comercial, política ou estratégica, da organização demandante. Apesar de a Engenharia de Software oferecer variados métodos para a gestão do DS, falhas de comunicação e tensões entre os grupos envolvidos são muito comuns.

Os relatos apresentados acima contêm temas percepções, narrativas, falas e materiais recolhidos de todas as áreas da Empresa (Gestão, Financeiro, Comunicação, Administrativo, RH), e da equipe de Desenvolvimento (Desenvolvedores, Gerente de Projetos, Engenheiro de Software e outros colaboradores da empresa). A empresa estudada, que, no início da pesquisa, trabalhava de uma forma familiar e artesanal, oferecendo produtos individualizados e personalizados para públicos especializados, viu seus negócios expandirem rapidamente e optou por ampliar seus mercados e produtos. Para isso, seus sócios entendiam que a melhor forma de se fazer isso era usando metodologias reconhecidas no mercado de desenvolvimento de software.

Os membros da empresa sentiam que era necessário inovar. Após dialogar com a equipe de DS e também com o setor financeiro da organização, foi decidido que a empresa empreenderia na criação de um novo software de gestão de bibliotecas com características diferenciadas dos já existentes no mercado. Esse software também significaria uma mudança importante em relação ao software de bibliotecas desenvolvido anteriormente pela empresa e que já era conhecido há mais de 10 anos pelo público especialista em Biblioteconomia e documentação. Porém junto com produto inovador, a empresa buscou também

modernizar o seu processo produtivo ao introduzir modelo MPS.Br como guia para a melhoria do desenvolvimento de software.

À época, o julgamento da gestão da empresa e também da equipe de DS, era que essa certificação possuía um peso considerável no mercado e que tê-la acabaria por incidir, tanto no currículo da empresa, como no currículo e qualificação dos Desenvolvedores que deveriam ser capacitados e qualificados para atender às exigências do modelo MPS.Br. Aliado a isso, a gestão já havia exposto para a equipe de DS sua insatisfação pelo fato dos sistemas construídos, até então, possuírem uma documentação básica de desenvolvimento, o que poderia ser resolvido, de acordo com os Desenvolvedores, pelo uso do modelo.

Para a empresa, o atendimento às condições propostas pelo MPS.Br pareceu complexo e exaustivo em um primeiro momento, mas, ao mesmo tempo muito coerentes na sua forma de aplicação e na proposição de seus objetivos. Por causa disso, a decisão pela implantação do modelo foi muito bem recebida pelas equipes, apesar de ter gerado algumas expectativas entre os atores envolvidos. Além da questão do status no mercado, a expectativa da Gestão era a de documentação, qualidade e rastreabilidade dos códigos desenvolvidos.

A expectativa do setor de DS era que o modelo determinasse um maior envolvimento da Gestão e da Especialista de Domínio no momento das escolhas de funcionalidades, e que esses tivessem um baixo grau de mudança de escopo de projeto durante sua execução. O Setor Financeiro e Gestão esperavam, também, que o tempo estimado e orçamentos destinados aos processos fossem cumpridos, e o Setor Comercial esperava ter em mãos, o mais rápido possível, um novo produto, o YesTeka, concluído e pronto para vendas.

Um dos pontos de tensão foi relativo aos comprometimentos que uma implementação com paradigmas para gerenciamento de produtos e gerenciamento e de requisitos exigiria. Na visão da Gestão e Diretorias, o maior grau de comprometimento deveria vir do envolvimento da equipe de DS relativos a prazos e qualidade dos códigos. Para a equipe de DS, um modelo desse porte só teria sucesso se a todas as instâncias da gestão se

comprometessem, incondicionalmente, na parte da aplicação do modelo que eles consideravam a mais importante: a criação e aprovação de todos os requisitos do software antes de iniciar o desenvolvimento.

No entanto, as expectativas de todos os envolvidos no movimento para a criação desse novo produto foram frustradas. De um lado, a implementação do MPS.Br não teve o êxito esperado em disciplinar a atividade dos Gestores e da Especialista de Domínio no que concernia à interferência na execução das tarefas após a definição das especificações. Do outro lado, o experimento de aumentar a distância entre a equipe de DS e os outros grupos da empresa acabou por contribuir para a piora da qualidade do software e a extensão dos prazos de execução.

Assim, com base na literatura e nos conceitos teóricos introduzidos no Capítulo 2, podemos identificar quatro principais mudanças na lógica organizacional promovidas pela introdução do MPS.br:

QUADRO 1 - LÓGICAS ORGANIZACIONAIS

	Antes do MPS.Br	Depois do MPS.Br
Lógica de produção	Artesanal	Industrial
Lógica de relação	Colaboração	Coordenação
Lógica de comunicação	Razão Comunicativa	Razão Instrumental
Lógica de resultado	Geração de produtos	Realização de tarefas

Fonte: Elaborado pela autora.

A introdução do MPS.Br modificou profundamente a lógica de produção na Empresa. Inicialmente, a forma como eram desenvolvidos os softwares se assemelhava à lógica de produção artesanal bem observada nos estudos de Marx (1964), com certo grau de curadoria e critérios de produção mais "artística" de códigos visando à satisfação do cliente que *"encomendava"* o produto. Como

mencionado em outros capítulos, os Desenvolvedores possuíam ampla autonomia na forma com que realizavam o desenvolvimento dos produtos. Além disso, era comum a realização de um módulo completo por um só Desenvolvedor, que podia imprimir seu próprio estilo e conhecimento ao trabalho realizado.

Por esse motivo, havia uma afinidade mais direta entre os Desenvolvedores e o próprio produto, de modo que eles se apropriavam dos resultados com um senso de orgulho e pertencimento. Isso fica claro no exemplo mencionado no capítulo 5, onde os Desenvolvedores chamavam os gestores e demais colaboradores da empresa para olharem, na tela de computador deles, resultados imediatos do seu trabalho.

Por outro lado, já durante a introdução do MPS.Br, instaurou-se uma forma de produção que se assemelha mais com lógica industrial. Existia por parte da empresa uma preocupação muito grande com a falta de padronização dos códigos gerados. Por causa disso, o caráter artesanal do desenvolvimento passou a ser visto, seja pelos programadores, seja pela gestão, como sinal de imaturidade da empresa. Por isso a busca por modelos que dessem ao setor o *"status"* de *"altamente profissionalizado"* foi muito privilegiada naquele momento.

De fato, com a introdução do MPS.Br o processo de desenvolvimento tornou-se mais fragmentado. O modelo instituiu uma distribuição de tarefas de curta duração para a equipe, ao invés de grandes blocos de código. O trabalho também passou a ser regulado de modo muito mais restrito pelo Engenheiro de Software, que agia como um integrador de um conjunto de tarefas paralelas que eram executadas pelos Desenvolvedores. Desse modo, havia a expectativa que seria possível gerar códigos mais padronizados, reaproveitáveis e com tempos de produção previsíveis.

Contudo, não foi isso o que aconteceu. Os programadores perderam a possibilidade de atuar como no passado, com maior criatividade e maior senso de apropriação. O que demonstra uma mudança na lógica de produção da

empresa, de uma lógica de criatividade para uma lógica tarefeira, muito explicitado no trabalho de Leal (2008) e Silva; Lima (2005). Subestimou-se a capacidade de fragmentar o trabalho em tarefas (tarefeiros *versus* produtos) e entre os membros da equipe que passaram a trabalhar de forma pontual e individualizada como se houvesse um entendimento velado que se cada um cumprisse a tarefa prescrita de acordo com o modelo, o software seria concluído em menor tempo e de forma satisfatória.

A partir daí tanto as consequências disso como as mudanças na lógica de produção observadas na empresa acabam por se assemelhar aos casos descritos pela literatura que analisa as contradições do desenvolvimento de softwares baseados apenas no uso de modelos de fabricas de software para o Desenvolvimento de Softwares (SILVA; LIMA, 2005; LEAL, 2008, CUKIERMAN; TEIXEIRA, 2007).

Nesse contexto, seja os conceitos positivos que negativos da empresa passaram a não ter responsáveis diretos, sinalizando dessa forma uma mudança também na lógica do resultado organizacional. Se por um lado, e em suas atividades anteriores, o foco da empresa e do setor de DS era a geração de produtos (no sentido amplo de planejamento e execução de um projeto), por outro, e após a introdução das metas de gerenciamento de produto e gerenciamento de requisitos exigidas pelo modelo MPS.Br, a relação do setor com o produto foi se distanciando cada vez mais e a responsabilidade apenas formal pelo resultado passou a prevalecer.

No que se refere à mudança da lógica das relações, a empresa, que até então trabalhava de maneira onde prevalecia a colaboração como forma de trabalho, vivenciou, após as mudanças processuais para a implantação do MPS.Br, um funcionamento com uma lógica de coordenação (RAJÃO, 2011), inclusive dando maior atenção às relações hierárquicas no setor de DS. Nesse sentido, a forma como as relações internas de hierarquia começaram a acontecer (com mais ocorrências de situações de coerção e coordenação que antes) acabaram por diminuir, também, as atividades de Comunidades de Prática que eram percebidas entre Desenvolvedores internamente e também com grupos externos.

Uma das características de Comunidades de Prática, de acordo com Wenger e Lave (1991), são as particularidades herméticas que esses grupos assumem quando em ação. Isso gera uma dificuldade para a relação entre comunidades de prática, como demonstrados em diversos exemplos citados nos capítulos anteriores. Porém, inicialmente, parte dessas dificuldades eram vencidas através da criação de vínculos pessoais e diretos entre a alta gestão, especialistas de domínio e os desenvolvedores. Com a introdução do MPS.Br ocorreu uma mudança na forma de coordenação, principalmente a partir da contratação de um Gerente de Projetos para fazer a ponte entre Desenvolvedores e gestão junto ao Engenheiro de Software.

Foi percebido que, já no momento da aquisição da consultoria para a implantação do MPS.Br o Engenheiro de Software, que trabalhava com a equipe em um modelo mais próximo à forma de colaboração do trabalho, estabeleceu hierarquias onde o grupo deveria atender às ordens e normas de trabalho vindas do próprio MPS.Br e também da gerência de projetos. Desse modo criou-se uma barreira entre os gestores e a equipe de DS.

Essa mesma situação acabou por influenciar, também, as lógicas das relações de comunicação da empresa. Ao meio do processo de implantação já se observava que as tarefas eram feitas de forma mais mecânica e com menos trocas entre os Desenvolvedores e as outras partes da empresa. O que antes funcionava via busca de formas de colaboração (aprendizado mútuo), múltiplos canais de comunicação, uso informal da linguagem acabou por migrar para um tipo de corporativismo e proteção mútua dentro do grupo de pratica. Esse processo se assemelha a uma passagem da razão comunicativa, defendida por Habermas (1987), para a predominância da razão instrumental de comunicação. Isso culminou em situações onde a alternativa da forma coercitiva do trabalho acabava por ser usada no lugar de formas mais colaborativas para cumprimento de tarefas urgentes da empresa.

Nesse momento, os grupos passaram a trazer informações esparsas e incompletas, de forma que nem a gestão e nem o Engenheiro de Software

tinham acesso ao todo informacional que levaria à solução ou enfrentamento do problema. Tal mudança não estava ligada à qualidade dos trabalhos ali desenvolvidos, mas sim, ao atendimento, mesmo que de forma inadequada ao que se estava sendo pedido pelo cliente. Isso acontecia, principalmente, no que se refere às mudanças de prazos e acertos de projeto.

Os dados apresentados acima sugerem que a mudança nas lógicas de produção e seus efeitos negativos têm relação com a modo determinístico com que o MPS.Br foi inicialmente visto pela empresa. Ao seguir de forma rígida e normativa a proposta de aplicação do MPS.Br, situações, tais como os atrasos nas entregas e escrita de uma modelagem do Banco de Dados que não previam flexibilizações e diálogo, foram sentidas em toda a empresa.

Isso acabou por interferir na dinâmica das atividades dos Desenvolvedores formando pontos de divergência que impactaram nas relações sociais, nas expectativas mercadológicas, econômicas, financeiras e, também, emocionais, dos atores envolvidos. Os processos hoje usados no DS vêm da década de 60 e percebemos que a contemporaneidade traz um colapso de inovação para os processos de criação formal, informal e corporativos (CERUZZI, 2006). No caso desse estudo, foi possível perceber isso ao verificarmos o grande grau de exaustão e insatisfação dos grupos ao sentirem que o produto desenvolvido totalmente dentro de uma metodologia reconhecida e testada não trouxe a qualidade e produtividade esperadas.

Tal resultado confirma a asserção de Leal, (2008), sobre os baixos índices de assertividade das organizações no que se refere ao sucesso no uso de formas tradicionais de DS. Entendemos que esse índice de insucesso o qual acreditamos ser maior que o divulgado por estudos e pesquisas devido ao grau de exposição que eles infligem a quem optou pelo uso desses modelos, possa ter como algumas de suas causas basais a falta de observância de que modelos tradicionais não dão mais conta de sobreviver em um mundo onde as relações de aprendizagem não privilegiam mais a alienação do trabalho por parte do empregador.

Esses modelos tradicionais disponíveis são, na maioria das vezes, voltados para a reprodução de ações previamente determinadas. Essa condução de aplicação de DS acaba por inibir os fluxos criativos dos atores por serem mais aderentes aos modelos do que entre as pessoas. Isso pode trazer uma sensação de desajustamento na medida em que na sociedade, a necessidade de aprendizagem, aplicação e verificação dos resultados do trabalho são parte inerente do desejo humano de criar e experimentar.

As evidências disponíveis nesse trabalho sugerem que a comunicação e as trocas especializadas são de grande importância para a melhoria da aprendizagem entre grupos sociais e que essas melhorias contribuem para avanços no mundo do trabalho. As revisões de literatura e abordagem teórica apresentados nessa dissertação corroboraram com essas evidências e foram de grande importância para um melhor entendimento sobre os desafios que a ES enfrentará para que haja avanços significativos de aprendizagem e produtividade na forma como os processos se dão no momento do DS.

A realização desse trabalho acabou por impactar tanto na vida profissional quanto na vida pessoal de muitos dos integrantes da empresa. A pesquisadora em especial viveu momentos delicados e contraditórios. De um lado, enquanto parte do processo do MPS.Br ela teve de ser fonte de motivação para a implantação do processo. Por outro lado, ter acesso à literatura crítica e compreender seu trabalho de forma diferente, passou a dar mais importância aos fatores sociais e organizacionais ligados ao DS. Para a pesquisadora, esses novos acessos ao conhecimento mudaram sua perspectiva sobre o tema inclusive incidindo em proposições vindas da parte dela que culminou em momentos de leitura e trocas coletivas que acontecem até hoje no setor de DS.

A Gestão também viu grande importância da pesquisa enquanto reflexão ao reconhecer que o trabalho próximo, relacional e coparticipativo já existia naquele ambiente e era desenvolvido antes. Foi demonstrado que apesar dessa forma de trabalho mais “artesanal” ter a pecha de “imaturo” ele possuia valores que agregavam em muito à forma como a empresa funcionava onde as pessoas se inter-relacionavam com o objetivo aprender e trazer soluções para o DS da

empresa. Houve ainda a reflexão empresarial de que nem sempre modelos amplos, feitos para atender uma grande gama de públicos é a resposta mais interessante para se chegar a melhorias, podendo escolhas desse tipo levar a empresa a situações delicadas e até mesmo perigosas para sua saúde financeira.

Fica a pergunta se, após a experiência de resultados não previstos, fazer a manutenção do uso do guia MPS.Br seria o reflexo do desejo de acerto desses grupos, ou, apenas, a conservação de uma escolha antecedente de alto custo financeiro e emocional. O que acontecerá futuramente na organização devido a essas escolhas ainda é incerto, porém, esse estudo forneceu elementos importantes para que sejam revistos elementos centrais para os processos e modelos de DS na empresa.

Este estudo também possui algumas implicações que podem ser levadas a outras organizações que atuam no setor de DS. Entendemos que, após a experiência por escolhas de mudança que afetaram as lógicas de produção, das relações, das comunicações e dos resultados, essa pesquisa possa contribuir para embasar decisões que envolvam ruptura do modo como uma empresa funciona uma vez que ficou a dúvida se haverá a manutenção da utilização do modelo tradicional de Desenvolvimento de Softwares, o MPS.Br.

REFERÊNCIAS

AGILE ALLIANCE. *Agile manifesto*. Disponível em < http://www.agilealliance.org >. Acesso em: 07 ago. 2015.

ALBUQUERQUE, João Porto de; SIMON, Edouard. *Desenvolvimento de software como desenho integrado de software e organização*. Workshop - Um Olhar Sociotécnico sobre a Engenharia de Software – WOSES. 5. [2009]. Disponível em: < http://www.uspleste.usp.br/jporto/woses2009/anais/02-WOSES-2009.pdf > Acesso em: 14 set. 2014.

ANTONELLO, Claudia Simone; RUAS, Roberto.*Formação gerencial*: pós-graduação lato sensu e o papel das comunidades de prática. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1415-65552005000200003> [2005]. Acesso em: 12 de dez. 2017.

BARRETT, Michael, OBORN, Eivor. *Boundary object use in cross-cultural software development teams*. Human Relations, 63(8), 1199-1221, 2010.

BROWN, John Seely; DUGUID, Paul. (1991). *Organizational learning and communities-of-practice*: toward a unified view of working, learning and innovation. Organization Science, v.2, 40-57.

BUTLER, Ricky; JOHNSON, Sally. Formal methods for life-critical software. Computing in Aerospace Conference. [9]. Disponível em < https://arc.aiaa.org/doi/pdf/10.2514/6.1993-4516 >. Acesso em 17 de jan. de 2017.

CARAVANTES, Geraldo R. *O ser total*: talentos humanos para o novo milênio. 2 ed. Porto Alegre: AGE, 2000.

CASTOR. Eduardo de Miranda. *Fábrica de Software: Passado, Presente e Futuro*. Disponível em: <http://www.unibratec.edu.br/tecnologus/wp-content/uploads/2006/08/n1_castor.pdf>. Acesso em: 02 ago. 2015.

CERUZZI, Paul E. *Storia dell'informatica*: dai primi computer digitali all'era di internet. Milano: Apogeo, 2006. 480 p.

CHRISSIS, Mary Beth; KONRAD, Mike; SHRUM, Sandy. *CMMI:* guidelines for process integration and product improvement. 2 ed. [s.n.t.]. 2007. 676 p. (SEI Series in Software Engineering).

CRAIG, Larman. *Utilizando UML e padrões*: uma introdução à análise e ao projeto orientados a objetos e ao desenvolvimento iterativo. São Paulo: Bookman, 2007. 695p.

CUKIERMAN, H.; TEIXEIRA, C. A. N. Por que Falham os Projetos de Implantação de Processos de Software. In: *Workshop - Um Olhar Sociotécnico sobre a Engenharia de Software* – WOSES. [2007]. 3. Porto de Galinhas. *Anais*.

CUKIERMAN, H.; TEIXEIRA, C. A. N.; R. PRIKLANDNICKI. *Um Olhar Sociotécnico sobre a Engenharia de Software*. RITA v.14 n.2. 2007. Disponível em < http://www.seer.ufrgs.br/index.php/rita/article/view/rita_v14_n2_p199-219/3547 >. Acesso em: 12 de jan. de 2015.

ENGESTROM, Yrjo et al. *Coordination, cooperation, and communication in the courts*: expansive transitions in legal work. In: COLE, M.; ENGESTROM, Y.; VASQUEZ, O. A. Mind, culture, and activity: seminal papers from the Laboratory of Comparative Human Cognition. Cambridge: Cambridge University Press. 1997.

ENGHOLM JR., Helio. *Engenharia de software na prática*. São Paulo: Novatec, 2010.

FAGUNDES, Eduardo. *Capability Maturity Model for software*. Disponível em <http://efagundes.com/artigos/capability-maturity-model-for-software/>. Acesso em: 15 de set. de 2016.

FERNANDES, Aguinaldo Aragon; TEIXEIRA, Descartes de Souza. *Fábrica de software*: implantação e gestão de operações. São Paulo: Atlas, 2007. 304 p. Disponível em: < http://www.mct.gov.br/upd_blob/0009/9238.pdf>. Acesso em: 02 set. 2015.

FERREIRA, R. B.; LIMA, F. P. A. Metodologias Ágeis: um novo paradigma de desenvolvimento de software. In: *Workshop Um Olhar Sociotecnico sobre a Engenharia de Software* – WOSES. [2005]. 3. Disponível em:<http://www.cos.ufrj.br/~handrade/woses/woses2006/pdfs/10-Artigo10WOSES-2006.pdf>. Acesso em: 10 jan. 2017.

FRANÇA, Alberto C., SILVA, Fábio Q. B. da. Um Estudo sobre Relações entre Papéis Funcionais do RUP e o Comportamento Pessoal no Trabalho em Equipe em Fábricas de Software. In: *WorkshopUm Olhar Sociotecnico sobre a Engenharia de Software* – WOSES. [2007]. 3. Disponível em: <https://www.researchgate.net/profile/Fabio_Silva19/publication/258023306_Um_estudo_sobre_Relacoes_entre_Papeis_Funcionais_do_RUP_e_o_Comportamento_Pessoal_no_Trabalho_em_Equipe_em_Fabricas_de_Software/links/54d49d600cf24647580607c4.pdf>. Acesso em: 28 jan. 2017.

GARAUDY, Roger. Karl Marx. Rio de Janeiro: Zahar, 1967.

GRAY, Barbara. *Collaborating*: finding common ground for multiparty Problems. San Francisco: Jossey Bass, 1989.

HABERMAS, JungenTeoria de la acción comunicativa.Trad. M.J. Redondo. Madrid: Taurus, 1987. 2.v.

HIRAMA, Kechi. *Engenharia de software*: qualidade e produtividade com tecnologia. Rio de Janeiro: Elservier, 2012. 209 p.

JERÔNIMO JUNIOR, Hélvio; CARVALHO, Rogério Atem de. Análise e viabilidade de aplicação do pensamento enxuto para atender os processos de produção de software do modelo de maturidade MPS.Br. In: Encontro Nacional de Engenharia de Produção. Desenvolvimento Sustentável e Responsabilidade Social: as Contribuições da Engenharia de Produção, 32.[2012].*Anais*. Disponível em: <http://www.abepro.org.br/biblioteca/enegep2012_TN_STO_158_922_20107.pdf> Acesso em: 10 dez. 2016.

LAPLANTE, Philip A. *What every engineer should know about software engineering*. New York: CRC, 2007. 312 p.

LEAL, L. *Saber social e desenvolvimento de software*: avaliação crítica do modelo da fábrica de software. Dissertação (Mestrado em Engenharia de Produção) – Escola de Engenharia, Belo Horizonte - UFMG, 2008.

LOBO, Edson J. R. *Guia prático de engenharia de software*. São Paulo: Digerati, 2009. 128 p.

MARTINS, José Carlos Cordeiro. *Gerenciando projetos de desenvolvimento de software com PMI*, RUP e UML. Rio de Janeiro: Compugraf, 2010. 283 p.

MARX, Karl. Manuscritos Econômico-Filosóficos. Lisboa: Edições 70, 1964.

NARDI, Julio Cesar; FALBO, Ricardo de Almeida. *Uma ontologia de requisitos de software.*(2006). Disponível em: <https://www.researchgate.net/publication/221561530_Uma_Ontologia_de_Requisitos_de_Software>. Acesso em: 23 nov. 2014.

OLIVEIRA, Ana Cláudia G. de; GUIMARÃES, Fernanda A.; FONSECA, Izabella. *Utilizando metodologias ágeis para atingir MPS.Br nível F*. Belo Horizonte: Softex, 2013. Disponível em: < http://www.softex.br/wp-content/uploads/2013/09/T1-PowerLogic-WE.pdf > Acesso em: 18 set. 2015.
PÁDUA FILHO, Wilson de. *Engenharia de software*: fundamentos, métodos e padrões. 2. ed. Rio de Janeiro: LTC, 2013.

POWELL, Walter W. (1990). *Neither marks nor hierarchy: network forms of organization*. In: B. M. Straw & L. L. Cummings. Research in organizational behavior. Greenwich, CT: JAI Press. v. 12, 295-336.
PRESSMAN, Roger S. *Engenharia de software*: uma abordagem profissional. 7. ed. Porto Alegre: Mc Graw Hill, 2011.

PRIKLADNICKI, R.; WILLI, R.; MILANI, F. *Métodos ágeis para desenvolvimento de software*. Porto Alegre: Bookman, 2014. 312p.
REZENDE, Denis Alcides. *Engenharia de Software e Sistemas de Informação*. 3 ed. Rio de Janeiro: BRASPORT, 2005.

ROBILLARD, Pierre N. The Role of Knowledge in Software. *Communications of the ACM*, v. 42, n. 1, 1999. p. 87-92.

RAJÃO, Raoni Guerra Lucas. *Objects, boundaries and joint work: the role of geographic information systems in the formulation and enforcement of deforestation control policies in Amazonia*. 2011. 286 f. Tese (Doutorado) - Curso de Filosofia, Department Of Organisation, Work And Technology, Lancaster University, Lancaster, 2011.

ROCHA, Ana Regina Cavalcanti da; MALDONADO, Jose Carlos; WEBER, Kival Chaves. *Qualidade de software*: teoria e prática. São Paulo: Prentice-Hall, 2001. 303 p.

SANDHOF, Karen; FILGUEIRAS, Lucia Vilela Leite. 75 Defeitos de Software como Erros Humanos. In: *Workshop Um Olhar Sociotécnico sobre a Engenharia de Software* - WOSES. [2006]. 2. Disponível em: <https://www.researchgate.net/profile/Lucia_Filgueiras/publication/266356181_Defeitos_de_Software_como_Erros_Humanos/links/5519338d0cf2d241f355c1f0.pdf>. Acesso em: 10 jan. 2016.

SANTOS JÚNIOR, Antônio F. dos; SANTOS, Rodrigo Pereira dos. *Workshop Um Olhar Sociotecnico sobre a Engenharia de Software* – WOSES. [2009]. 5. Disponível em: < http://eloquium.com.br/wp/wp-content/uploads/2016/05/ArtProva2.pdf> Acesso em: 08 jan. 2016.

SCHACH, Stephen R. *Engenharia de software*: os paradigmas clássico e orientado a objetos. São Paulo: Mc Graw Hill, 2010. 617 p.

SILVA, A. L.; LIMA, F. P. A. Análise de Requisitos de Software e Análise da Atividade de Trabalho. In: *Workshop Um Olhar Sociotécnico sobre a Engenharia de Software* – WOSES. [2005] 1. Rio de Janeiro. Anais.

Site Oficial do CMMI. *Maturidade e desenvolvimento de software*. Disponível em ,<http://www.devmedia.com.br/maturidade-no-desenvolvimento-de-software-cmmi-e-mps-br/27010>. Acesso em 30 fev. de 2017

SOFTWARE ENGINEERING INSTITUTE. *Watts Humphrey, un outrageous commitment, a lifelong mission*. Disponível em: https://www.sei.cmu.edu/watts/. Acesso em: 21 maio de 2014.

SOMMERVILLE, Ian. *Engenharia de software*. 8 ed. São Paulo: Cambrige: Addison Wesley, 2008. 552 p.

TEIXEIRA, C.A.N. *Um olhar sociotécnico sobre a Engenharia de software: o caso do BNDES*. Dissertação de mestrado pela Engenharia das Ciências e Computação: COPPE/UFRJ, 2007. 168 p. Disponível em < https://web.bndes.gov.br/bib/jspui/handle/1408/10254 >. Acesso em: 8 de dez. de 2014.

THE STANDISH GROUP INTERNATIONAL. *CHAOS MANIFESTO 2013 Think Big, Act Small*. Boston: The Standisch Group International, 2013. Disponível em: <

http://www.immagic.com/eLibrary/ARCHIVES/GENERAL/GENREF/S130301C.pdf>. Acesso em: 12 out. 2013.

THOMAS, R; SARGENT L; HARDY, C. *Power and participation in the production of boundary objects*. Working Paper Series, Cardiff Business School. 2008.

VERAS, Marcelo. *Inovação e métodos de ensino para nativos.* São Paulo: Atlas, 2011. 156 p.

WAZLAWICK, Raul Sidnei. *Engenharia de software*: conceitos e práticas. Rio de Janeiro: Elsevier, 2013.

WENGER, Etienne.*Comunità di prática*. Apprendimento, significato e identità. Milano: Cortina Raffaello, 2006. 386 p.

WENGER, Etienne; LAVE, Jean. Situated Learning: Legitimate Peripheral Participation (*Learning in Doing: Social, Cognitive and Computational Perspectives*).Cambridge: Cambridge University Press, 1991. 198 p.

WENGER, Etienne; McDERMOTT, Richard; SNYDER, William. *Coltivare comunità di pratica*: Prospettive ed esperienze di gestione della conoscenza. Milano: Guerini e Associati, 2007. 292 p.

WENGER, Etienne;WENGER, Beverly. *Introduction to communities of practice: a brief overview of the concept and its uses*. Disponível em: < http://wenger-trayner.com/introduction-to-communities-of-practice/ >. Acesso em: 01 mar. 2017.