CARLOS EDUARDO CHESSI MELO SILVA

ESTÁGIO EM DESENVOLVIMENTO WEB NO LEMAF/UFLA

LAVRAS - MG

2014

CARLOS EDUARDO CHESSI MELO SILVA

ESTÁGIO EM DESENVOLVIMENTO WEB NO LEMAF/UFLA

Relatório de estágio apresentado na Universidade Federal de Lavras – UFLA como requisito para obtenção do título de graduação no curso de Sistemas de Informação, sob orientação do Professor Joaquim Quinteiro Uchôa.

LAVRAS - MG

2014

CARLOS EDUARDO CHESSI MELO SILVA

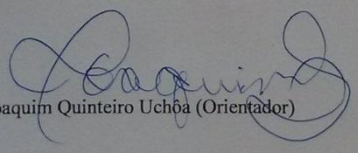
ESTÁGIO EM DESENVOLVIMENTO NO LEMAF/UFLA

Monografia de graduação apresentada ao Colegiado do Curso de Bacharelado em Sistemas de Informação, para obtenção do título de Bacharel.

APROVADA em 10 de julho de 2014.

Hermes Pimenta de Moraes Júnior

Fernando Simeone



Joaquim Quinteiro Uchôa (Orientador)

LAVRAS-MG
2014

ESTÁGIO EM DESENVOLVIMENTO WEB NO LEMAF/UFLA

Autor: Carlos Eduardo Chessi Melo Silva

Orientador: Joaquim Quinteiro Uchôa

Resumo: Este relatório de estágio descreve as atividades desenvolvidas no Laboratório de Projetos e Estudos em Manejo Florestal – LEMAF. O estágio abordado teve como base o desenvolvimento de *software* de gestão integrada voltado ao setor ambiental. Serão apresentadas descrições das atividades desenvolvidas, as tecnologias empregadas, metodologias e os processos de atividades que foram desempenhadas na área de Tecnologia de Informação durante o período de estágio, destacando as metodologias ágeis, *XP* e *Scrum,* as quais o LEMAF adota para o desenvolvimento dos sistemas *web*. Assim, o trabalho relata que o estágio proporciona qualificação técnica, permitindo a prática de trabalhar em equipe, respeitando as diferenças e visualizando os benefícios que a prática ocasiona.

Palavras-chave: Metodologias Ágeis, *Scrum*, *XP,* Desenvolvimento *web.*

LISTA DE ILUSTRAÇÕES

SUMÁRIO

1. INTRODUÇÃO

O presente trabalho tem por finalidade relatar experiências vividas durante o estágio realizado no LEMAF (Laboratório de Projetos e Estudos em Manejo Florestal), que se encontra no Departamento de Ciências Florestais – DCF da Universidade Federal de Lavras – UFLA. O laboratório foi criado em 2002 e desde então conduz diversos projetos em parceria e ou convênio com órgãos estaduais e federais e iniciativa privada, tendo como objetivos efetuar pesquisa, ensino e extensão.

As principais áreas de atuação do LEMAF são: Manejo e Inventário Florestal, Sensoriamento Remoto, Geoprocessamento, Economia Florestal, desenvolvimento de Sistemas de Informação Geográficos e de Sistemas de Informação voltados à Gestão Ambiental.

O laboratório é constituído por aproximadamente 200 pessoas, de modo que, 60 funcionários e 20 estagiários estão diretamente vinculados à área de Tecnologia de Informação, tendo como supervisor e responsável Samuel Rodrigues de Sales Campos.

O estágio abordado nesta teve como base o desenvolvimento de *software* de gestão integrada voltado ao setor ambiental, no qual são essenciais o trabalho em equipe e a divisão de responsabilidades. Deste modo, o objetivo do estágio foi proporcionar a utilização das melhores tecnologias (ferramentas e *frameworks*) no desenvolvimento dos sistemas *web* que requerem conhecimentos nas áreas de Engenharia de Software, Linguagens de Programação e Banco de Dados.

O desenvolvimento de *software* dos projetos envolvidos no estágio foi realizado utilizando-se metodologias de desenvolvimento conhecidas como XP (*Extreme Programming*) e *framework Scrum*. Tais metodologias são usadas para proporcionar o desenvolvimento ágil dos projetos e facilitar o trabalho dos desenvolvedores.

Como os projetos desenvolvidos no LEMAF têm características de rotatividade de pessoas nas equipes, o plano de trabalho torna-se dinâmico de acordo com a necessidade dos projetos e das equipes onde o estagiário ficará alocado. Este estágio, especificamente, é um importante instrumento para o futuro dos profissionais na área de Tecnologia da Informação, pois

através dele é possível aprimorar técnicas, ajudando o estagiário a aprender como funciona a dinâmica do desenvolvimento de *software* em uma situação real e a lidar com diferentes desafios encontrados no dia a dia de um profissional.

Neste contexto, estagiar pelo LEMAF tornou-se motivador, pelo fato de oferecer oportunidades de adquirir conhecimento na área da Tecnologia de Informação, e consequentemente experiências que serão colocadas em prática no mercado de trabalho. Contudo, espera-se que haja uma absorção de tudo que está sendo abordado no estágio e proporcione um amadurecimento em relação a conhecimento, produtividade e interação dentro do ambiente de trabalho.

2. REFERENCIAL TEÓRICO

Visto que o estágio abordado teve como base o desenvolvimento de *software*, é necessário que alguns conceitos sejam definidos e analisados através de referenciais. Estes proporcionarão maior clareza sobre as ferramentas e abordagens utilizadas no LEMAF.

Como mencionado anteriormente, o LEMAF desenvolve projetos voltados para *web*. Pode-se definir uma aplicação *web* como uma aplicação de *software* que utiliza a *web* como ambiente de execução. Aplicações *web* envolvem sites *web* ou sistemas *web* (CONALLEM, 2000).

O desenvolvimento *web* é caracterizado pela constante pressão por prazos cada vez menores, num ambiente onde a competição depende da velocidade no atendimento às necessidades dos usuários e do negócio (CHAUBEY, 2001). Assim, o desenvolvimento *web* faz com que os sistemas de *software* sejam desenvolvidos cada vez mais rápidos, de forma eficaz e a um baixo custo. Consequentemente o cliente deseja que o *software* funcione sempre e de forma correta e, em caso de falha, ele espera que o desenvolvedor rapidamente corrija o erro, independente do grau de dificuldade do motivo da falha (FENTON & PFLEEGER, 1997).

Para minimizar toda essa pressão, competitividade e eficiência em um ambiente de desenvolvimento de *software*, uma corrente crescente de novas metodologias vem tomando corpo nos últimos anos. Estas novas metodologias de desenvolvimento denominadas de Métodos Ágeis (FOWLER, 2000b), baseiam-se em conceitos e práticas já conhecidas. Muitas dessas práticas são herdadas da Engenharia de *Software* tradicional, unidas de forma diferenciada, num processo mais adequado com a visão e necessidades atuais (ABRAHAMSSON *et al*, 2003).

Ainda segundo este autor, o advento dos Métodos Ágeis de desenvolvimento de *software* é uma resposta clara à pressão imposta ao desenvolvimento de *software*, apresentando um conjunto de processos, atividades e papéis que se propõe a entregar *software* de qualidade no menor prazo possível. Esses métodos, entretanto, abrangem, parcialmente, as peculiaridades do desenvolvimento de sistemas *web*, deixando lacunas a

serem preenchidas por empresas e equipes focadas no desenvolvimento de sistemas *web*.

Os Métodos Ágeis constituem uma nova classe de metodologias de desenvolvimento de *software* criada para atender à crescente pressão do mercado por processos mais ágeis e leves, com ciclos de desenvolvimento cada vez mais curtos. Essas características são particularmente marcantes no volátil e crescente mercado de sistemas *web*, assim como no emergente ambiente de aplicações móveis (ABRAHAMSSON *et al,* 2003).

As principais metodologias que hoje são conhecidas como Métodos Ágeis surgiram por volta da década de 90, período de crescimento do uso comercial da *web*. Destacam-se entre estas metodologias: *Extreme Programming* (XP) (BECK, 2000), *Crystal Methods* (COCKBURN, 2002), *Lean Development* (HIGHSMITH, 2002), *Feature-driven Development* (FDD) (PLAMER & FELSING, 2002), *Adaptive Software Development* (ASD) (HIGHSMITH, 2000), *Scrum* (SCHWABER & BEEDLE, 2002) e *Dynamic System Development Methodology* (DSDM) (HIGHSMITH, 2002).

Somente a partir de 2001, com a publicação do Manifesto para o Desenvolvimento Ágil de *Software* (MANIFESTO, 2001), os Métodos Ágeis passaram a ser conhecidos como tal. O Manifesto foi o resultado de uma conferência onde os principais líderes e proponentes das metodologias supracitadas se reuniram para discutir as semelhanças e diferenças de suas abordagens para o desenvolvimento de *software* tradicional (HIGHSMITH, 2001).

Os Métodos Ágeis não são aplicáveis a qualquer domínio. A utilização de métodos ágeis em equipes desmotivadas ou em organizações centradas em processo, não é aconselhável. Igualmente desaconselhável é a adoção de Métodos Ágeis em projetos onde a colaboração do cliente é limitada ou com equipes muito grandes. O tamanho médio das equipes utilizando métodos ágeis é de nove pessoas, embora existam exemplos de projetos bem sucedidos com até 250 pessoas (COCKBURN & HIGHSMITH, 2001).

Segundo Cockburn & Highsmith (2001) a maioria das metodologias ágeis não possui nada de novo. O que as diferencia das metodologias tradicionais são o enfoque e os valores. A ideia das metodologias ágeis é o enfoque nas pessoas e não em processos. Além disso, existe a preocupação de gastar

menos tempo com documentação e mais com a implementação. Uma característica das metodologias ágeis é que elas são adaptativas ao invés de serem preditivas. Com isso, elas se adaptam a novos fatores decorrentes do desenvolvimento do projeto, ao invés de procurar analisar previamente tudo o que pode acontecer no decorrer do desenvolvimento (SOARES, s.d).

Ainda segundo Soares, apesar do uso crescente das metodologias ágeis, ainda falta uma base maior de projetos para verificar suas vantagens. Mesmo assim, os resultados iniciais em termos de qualidade, confiança, data de entrega e custo são promissores. Para ser realmente considerada ágil, a metodologia deve aceitar a mudança ao invés de tentar prever o futuro. O problema não é a mudança em si, mesmo porque ela ocorrerá de qualquer forma. O problema é como receber, avaliar e responder às mudanças.

Enquanto as metodologias ágeis variam em termos de práticas e ênfases, elas compartilham algumas características, como desenvolvimento iterativo e incremental, comunicação e redução de produtos intermediários, como documentação extensiva. Desta forma existem maiores possibilidades de atender aos requisitos do cliente, que muitas vezes são mutáveis (SOARES, s.d).

As metodologias ágeis *Scrum* e XP serão definidas e analisadas particularmente para melhor compreensão, visto que durante o estágio estas merecem destaque por serem mais eficazes e eficientes na execução das tarefas realizadas no LEMAF.

2.1. *Extreme Programming* (XP)

A *Extreme Programming,* mais conhecida como XP, é uma metodologia ágil para desenvolvimento de software, e tem como principais objetivos entregar funcionalidades de forma rápida e eficiente ao cliente.

Sua utilização é recomendada em projetos com requisitos incertos e que possuam equipes pequenas (máximo de doze desenvolvedores), no qual o sistema possa ser desenvolvido de forma incremental (ou iterativa) e que a programação seja feita usando o paradigma da orientação a objetos (RAMOS, 2013).

Na XP o foco principal é o escopo, este prioriza o desenvolvimento das funcionalidades que agreguem o maior valor para o negócio do cliente. Para isso a equipe deve reunir todas as habilidades técnicas e de negócios para produzir o *software* (BASSI FILHO, 2008). Sendo assim, a metodologia enfatiza o desenvolvimento rápido do projeto, além de favorecer o cumprimento das estimativas.

O ciclo de planejamento e *feedback* da metodologia, devem ser diários. A equipe deve se reunir e discutir sobre os resultados e dificuldades encontradas até o momento, priorizar aquilo que é mais importante para o cliente e focar seus esforços para entregar *software* funcionando. Desta forma, as principais diferenças da metodologia XP com relação as demais são: o *feedback* constante; abordagem incremental; e a comunicação entre as pessoas é encorajada (SOARES, s.d).

A XP é uma forma nova de tratar o relacionamento entre clientes do desenvolvimento de software e os desenvolvedores. Clientes e desenvolvedores se envolvem em uma espécie de dança, ambos os lados trabalhando juntos para o bem de toda a equipe. (TELES, 2004, p. 297).

Contudo, a XP é ideal para ser usada em projetos em que os *stakeholders*[1] não sabem exatamente o que desejam e podem mudar muito de opinião durante o desenvolvimento do projeto. Com *feedback* constante, é possível adaptar rapidamente eventuais mudanças nos requisitos (SOARES, s.d).

2.2. *Scrum*

A metodologia *Scrum* geralmente é composta por times trabalhando como uma unidade altamente integrada com cada membro desempenhando um papel bem definido e o time inteiro focando num único objetivo – entrega do produto. Tal metodologia elimina práticas de controle desnecessárias, inadequadas e burocráticas, concentrando-se na essência do processo de confecção de sistemas de informação (FONSECA, 2009).

---

[1] Representam as partes interessadas do projeto.

A *Scrum* é bastante objetiva, possuindo metas claras, equipe bem definida, flexibilidade, comprometimento e cooperação do time; e sua curva de aprendizado é relativamente baixa (MACHADO & MEDINA, s.d).

O foco da metodologia é encontrar uma forma de trabalho dos membros da equipe para produzir o *software* de forma flexível e em um ambiente em constante mudança. Isso comprova a afirmação de Schwaber (2004), que os pontos fortes da *Scrum* são: “Adaptabilidade e Flexibilidade”.

Na metodologia *Scrum*, o *Product Owner* é a pessoa que tem o papel de maior responsabilidade e visibilidade dentro do projeto, pois é ele que irá realizar a comunicação entre o cliente e a equipe de desenvolvimento. Com o desenvolvimento ágil, cada requisito do *software* é dividido em atividades e dispostas em um *kanban*[2]. Cada desenvolvedor escolhe a atividade que queira realizar. Nesse ponto é importante o papel do *Scrum Master*, pois é sua responsabilidade ter a percepção de possíveis dificuldades encontradas por cada membro da equipe, ajudando o desenvolvedor como for possível. Desta forma, a Figura 1 representa o *kanban*.

Figura 1 – *Kanban.*

[2] É um conceito relacionado com a utilização de cartões (post-it e outros) para indicar o andamento da *sprint*.

O ciclo da *Scrum* tem o seu progresso baseado em um série de iterações bem definidas, cada uma com duração de 15 dias, chamadas *sprints*[3]. Antes de cada *sprint*, realiza-se uma reunião de planejamento (*Sprint Planning*) onde a equipe (*Scrum Team*) de desenvolvedores tem contato com o *Product Owner* para priorizar o trabalho que precisa ser feito, selecionar e estimar (*Planning Poker*) as funcionalidades que o time pode realizar dentro da *sprint*. A próxima fase é a execução da *sprint*. Durante a execução da *sprint*, o time controla o andamento do desenvolvimento realizando reuniões diárias, observando o seu progresso usando um gráfico chamado *burndown* (PEREIRA, P. *et al* 2007*).*

Ao final de cada *sprint*, é feita uma revisão no produto entregue para verificar se tudo realmente foi implementado. Deve-se realizar uma reunião de revisão (*Sprint Review*), onde o time demonstra o produto gerado na *sprint* e valida se o objetivo foi atingido. Logo em seguida, realiza-se a reunião de retrospectiva (*Sprint Retrospective*), uma reunião de lições aprendidas, com o objetivo de melhorar o processo/time e/ou produto para a próxima *sprint* (PEREIRA, P. *et al* 2007*).* Desta forma, a Figura 1 representa todo o processo da metodologia *Scrum* explicado anteriormente.

[3] Ciclo que representa um tempo definido, dentro do qual um conjunto de atividades deve ser executado.

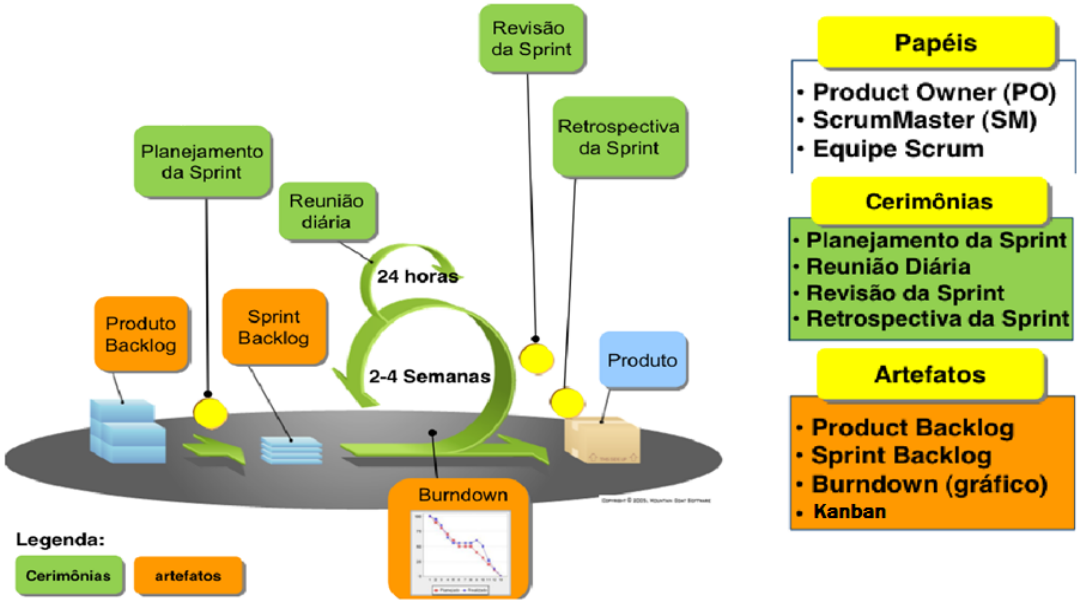


Figura 2 - Processo da metodologia *Scrum.*

Segundo este mesmo autor, a *Scrum* torna-se ideal para projetos dinâmicos e suscetíveis a mudanças de requisitos, sejam eles novos ou apenas requisitos modificados. No entanto, para aplicá-lo, é preciso entender antes os seus papéis, responsabilidades, conceitos e artefatos das fases de seu ciclo.

A metodologia *Scrum,* de certa maneira, é baseada em princípios semelhantes aos da XP: equipes pequenas, requisitos pouco estáveis ou desconhecidos e iterações curtas para promover visibilidade para o desenvolvimento (MACHADO & MEDINA, s.d).

2.3 Tecnologias

Nesta seção, serão feitas descrições das linguagens e *frameworks* utilizadas no LEMAF.

2.3.1 Linguagens de Programação

2.3.1.1 Java

Java é uma linguagem computacional completa, adequada para o

desenvolvimento de aplicações baseadas na rede Internet, redes fechadas ou ainda programas stand-alone [4](CAMPIONE, M; WALRATH, K. 2006).

2.3.1.2 HTML

HTML é uma linguagem com a qual se definem as páginas web. Basicamente trata-se de um conjunto de etiquetas (tags) que servem para definir a forma na qual se apresentará o texto e outros elementos da página. O HTML se criou a princípio com objetivos de divulgação. Porém, não se pensou que a web chegaria a ser uma área de ócio com caráter multimídia, de modo que, o HTML se criou sem dar respostas a todos os possíveis usos que lhe dariam posteriormente e a todo coletivo de gente que o utilizariam no futuro (ALVAREZ, 2004).

2.3.1.3 JavaScript

JavaScript é uma linguagem que permite injetar lógica em páginas escritas em HTML (HiperText Mark-up Language). Os parágrafos de lógica do javaScript podem estar "soltos" ou atrelados a ocorrência de eventos. Os parágrafos soltos são executados na sequência em que aparecem na página (documento) e os atrelados a eventos são executados apenas quando o evento ocorre (JUNIOR, s.d).

2.3.1.4 CSS

O Cascading Style Sheets (CSS) é uma "folha de estilo" composta por “camadas” e utilizada para definir a apresentação (aparência) em páginas da internet que adotam para o seu desenvolvimento linguagens de marcação (HTML). O CSS define como serão exibidos os elementos contidos no código de uma página da internet e sua maior vantagem é efetuar a

[4] Trata-se de uma aplicação independente

separação entre o formato e o conteúdo de um documento (PEREIRA, 2009).

2.3.3 Frameworks

Para Fayad (1999b) e Johnson & Foote (1988), um *framework* é um conjunto de classes que constitui um projeto abstrato para a solução de uma família de problemas.

2.3.3.1 Hibernate

O Hibernate é um *framework* ORM - Object Relational Mapping. É uma ferramenta que nos ajuda a persistir objetos Java em um banco de dados relacional. O trabalho do desenvolvedor é definir como os objetos são mapeados nas tabelas do banco e o Hibernate faz todo o acesso ao banco, gerando inclusive os comandos SQL necessários (CAELUM, 2014).

2.3.3.2 Spring MVC

O Spring MVC, uma das partes do *framework* Spring, é um *framework* no estilo web maduro e capaz de responder as ações, com uma grande variedade de recursos e opções destinadas a manipulação de vários casos de usos focados ou não na web. Todos esses recursos podem tornar a vida do neófito bastante difícil (MOTA, 2006).

2.3.3.3 PlayFramework

Segundo Bridee (2011), o Play é um *framework* open source para aplicações web, escrito em Java, que possibilita o desenvolvimento de aplicações web que seguem o padrão MVC. Tem por objetivo otimizar a produtividade do desenvolvedor através do uso de configuração sobre convenção (CoC).

2.3.3.4 Swiz

Swiz é um simples *framework* de inversão de controle. Usando um *framework* IoC, os componentes de sua aplicação (por exemplo, Views) não instanciam ou mesmo olham para suas dependências (os objetos com os quais eles trabalham). O *framework* injeta estas dependências quando os componentes são criados. O resultado é baixo acoplamento e componentes mais reutilizáveis (ROSE, 2009).

2.3.3.5 JQuery

JQuery é uma poderosa biblioteca JavaScript criada para simplificar a criação de efeitos visuais e de interatividade em web sites. Desenvolvedores especialistas em JavaScript, ao conhecerem as maravilhas de que a biblioteca é capaz. JQuery propicia a criação de scripts de uma forma tão simples e intuitiva. Simplicidade foi a diretriz que norteou John Resig na criação da biblioteca (SILVA, 2008).

2.3.3.6 Angular JS

AngularJS é um *framework* do time de desenvolvedores do Google. Diferentemente de outros frameworks JavaScript, ele adota uma abordagem mais ligada à sintaxe HTML, funcionando como uma espécie de extensão da linguagem (FERREIRA, 2012).

2.3.3.7 Mootools

O Mootools é voltado para o desenvolvedor com conhecimentos intermediários e avançados em JavaScript e permite escrever códigos mais poderosos e flexíveis com elegância, além disso, o Mootools é um framework bem documentado e com uma API coerente que vem de encontro aos anseios dos desenvolvedores mais experientes(CHUMA, s.d).

3. METODOLOGIA

Nesta etapa do presente trabalho, serão apresentadas descrições sobre as atividades desenvolvidas, as tecnologias empregadas e os ramos de atividades que foram desempenhadas durante o período de estágio no LEMAF.

Os projetos desenvolvidos pelo LEMAF são sistemas *web* e tem como base o desenvolvimento de *software* de gestão integrada voltados ao setor ambiental. Portanto, consistem basicamente de uma aplicação rodando no lado do servidor e outra que roda no lado do cliente, que no caso é um *browser*. Desta forma, os projetos trabalhados, foram desenvolvidos tanto no *backend* (servidor) quanto no *frontend* (cliente).

No desenvolvimento da aplicação do servidor, o LEMAF trabalha com Java e algumas tecnologias de apoio ao desenvolvedor como *frameworks*. Teve-se a oportunidade de aplicar alguns padrões como MVC[5] e trabalhar com a linguagem Java, *frameworks* como Hibernate, Spring MVC, PlayFramework. No desenvolvimento das aplicações clientes, que rodam no *browser*, o LEMAF trabalha com linguagens como: Flex, JavaScript, Html e CSS, e *frameworks* como Swiz, JQuery, Angular JS e Mootools.

Durante todo período de estágio, trabalhei em três projetos diferentes e em equipes diferentes, pois existe uma grande rotatividade das equipes. O primeiro projeto foi Fiscalização Ambiental, que tem como finalidade gerenciar todas as Denúncias e Demandas do Ministério Público de caráter ambiental, em que será possível manter o contato com o cidadão de forma a apurar/orientar as solicitações. As demandas são recebidas via site ou atendimento telefônico, o sistema provê uma entrada de dados através de uma interface utilizada pelos atendimentos do *call center*. A Figura 3 representa o site, o qual o cidadão irá realizar as denúncias.

[5] (Modelo Visualização Controle) fornece uma maneira de dividir a funcionalidade envolvida na manutenção e apresentação dos dados de uma aplicação.

Figura 3 - Site Fiscalização Ambiental.

Desta forma, durante esse projeto atuei como desenvolvedor, de modo que foram desenvolvidos dois módulos: um para usuários internos e o outro (site) para usuários externos. No módulo interno trabalhei com as linguagens Java, Flex e *frameworks* como Swiz e Spring MVC. Já no externo com Java, HTML, CSS e JavaScript utilizando ferramentas de apoio como JQuery, Mootools.

O segundo projeto está em desenvolvimento, Regularização Ambiental – Certidão de Dispensa, que é a primeira etapa prevista para o Programa de Regularização Ambiental (PRA) do estado de Minas Gerais, conforme ilustra a Figura 4.

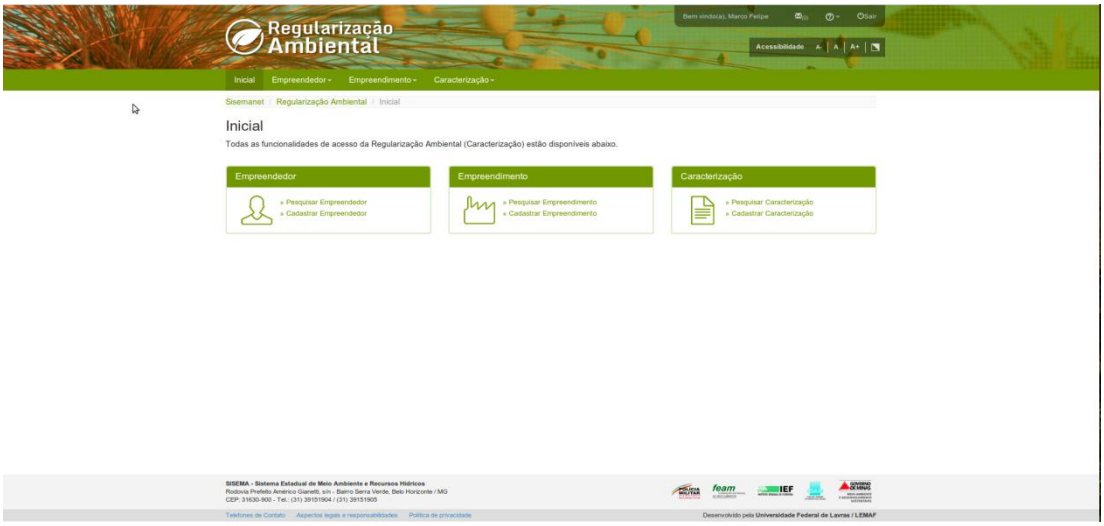


Figura 4 - Regularização Ambiental.

E o terceiro é Projeto Malha - Urubu, que tem como objetivo apontar possíveis pontos de atropelamento de animais silvestres e assim como classificá-los. Tal projeto pode ser visto na Figura 5.

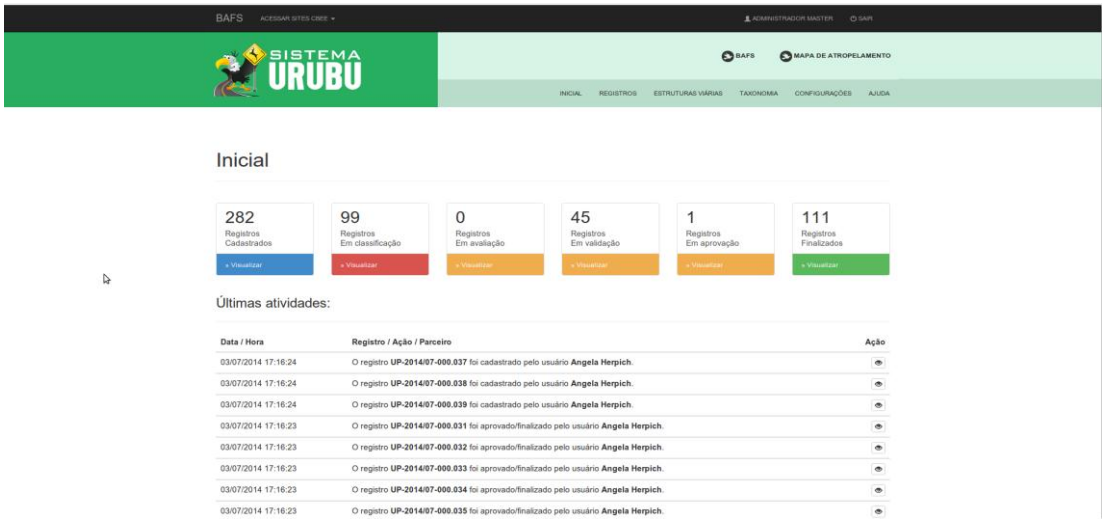


Figura 5 - Projeto Malha - Urubu.

Nos dois últimos projetos, foram utilizadas as mesmas tecnologias: Java, PlayFramework, HTML, CSS, JavaScript, JQuery e AngularJS. A qual também atuei como desenvolvedor.

Em todo projeto desenvolvido pelo LEMAF, realizou-se uma reunião na qual estavam presentes o *Product Owner*, o *Scrum Master* e qualquer pessoa interessada que estivesse representando o cliente. Durante esta reunião, o *Product Owner* descreveu as funcionalidades de maior prioridade para o desenvolvimento do projeto. Tais funcionalidades deram origem ao *backlog*[6].

Após o levantamento do *backlog,* as atividades foram planejadas e filtradas para serem desenvolvidas em um prazo determinado por uma *sprint*, conhecida como *Sprint Planning.* Esta etapa foi composta pelo *Product Owner, Scrum Master* e *Scrum Team*.

Durante a *Sprint Planning*, ocorria uma prática de estimativa de tarefas em horas bem simples e muito eficiente, composta pelo *Scrum Master* e

[6] Representa tudo o que deverá ser feito durante a próxima Sprint do seu projeto. Ele surge a partir do que foi levantado e listado, pelo Product Owner.

Scrum Team, conhecida como *Planning Poker.* Assim, consequentemente, acontecia uma *sprint*, que consiste em um grupo de funcionalidades definidos em um *backlog*. Uma *sprint* durava em torno de 15 dias.

No decorrer da *sprint,* foram realizadas reuniões diárias onde cada membro do time falou a respeito da atividade na qual ele está trabalhando, deixando toda a equipe a par de como está indo o desenvolvimento do projeto como um todo, observando o seu progresso usando artefatos como *kanban e* um gráfico chamado *burndown.* O sucesso da *sprint* foi avaliado mais adiante na *Sprint Review* em relação ao objetivo traçado para a *sprint*.

Assim, para identificar o que funcionou bem, o que pode ser melhorado e que ações foram tomadas para melhorar, foi necessário a *Sprint Retrospective* que ocorre ao final, como uma retrospectiva da *sprint*.

O que normalmente mudou de um projeto para outro foi somente a regra de negócios, que é inerente ao projeto, e também as tecnologias, que diferem dependendo das particularidades de cada projeto.

A equipe de estagiários do LEMAF, em geral, realiza desenvolvimento de *software* utilizando práticas do XP que consiste em um processo de desenvolvimento de *software* que adota os valores de comunicação, simplicidade, *feedback* e coragem. Estes quatro valores servem como critérios que norteiam as pessoas envolvidas no desenvolvimento de *software* e facilitam a absorção de novos conhecimentos e experiências. Muitas vezes o desenvolvimento é realizado em pares para que a comunicação entre os desenvolvedores seja facilitada.

No gerenciamento do desenvolvimento ágil dos projetos, foi também utilizada a metodologia *Scrum*. Os times *Scrum*, como são chamadas as equipes de desenvolvimento no *Scrum*, são auto-organizáveis, multidisciplinares e idealizadas. Cada membro do time tem seu papel bem definido sendo que o *Scrum Master* é quem orienta e gerencia os processos do desenvolvimento de um determinado time, podendo também ser um desenvolvedor, conforme o restante que compõe a equipe.

Conforme descrito e vivenciado no estágio, os desenvolvimentos dos projetos que utilizaram as metodologias XP e Scrum, conjuntamente, proporcionaram melhores resultados em termos de cumprimento de prazos e padrões de qualidade nos projetos desenvolvidos no LEMAF.

Contudo, é perceptível que o LEMAF utiliza-se uma vasta variedade de tecnologias no desenvolvimento dos projetos o que proporciona uma multidisciplinaridade de conhecimento. Desta forma, foi possível aprender na prática um pouco mais sobre como desenvolver um sistema distribuído e sobre questões relacionadas a arquitetura de um sistema, tendo assim uma noção sobre qual tecnologia deve ser utilizada e quais as vantagens, desvantagens e limitações das ferramentas e tecnologias utilizadas durante esse período de estágio.

4. CONCLUSÃO

Com toda base adquirida durante o curso de Sistemas de informação, ao término do estágio realizado no LEMAF, chega-se à conclusão de que ele foi de grande valia para concretizar todo o aprendizado exposto no trabalho.

Através do estágio foi possível ter uma visão ampla das metodologias, XP e Scrum, as quais proporcionam melhores resultados em termos de cumprimento de prazos e padrões de qualidade nos projetos desenvolvidos no LEMAF.

O estagiário do LEMAF aprende na prática um pouco mais sobre como desenvolver um sistema de *software*, utilizando uma ampla variedade de tecnologias no desenvolvimento dos projetos. O estágio proporciona também uma noção sobre qual tecnologia deve ser utilizada e quais os pontos positivos, negativos e obstáculos das ferramentas e tecnologias utilizadas durante esse período de estágio. Tais práticas exercidas no LEMAF fazem com que o estagiário adquira uma multidisciplinaridade de conhecimento.

Assim, o programa de estágio favoreceu a qualificação técnica, emocional e social, permitindo a prática de trabalhar em equipe, respeitando as diferenças e visualizando os benefícios que a prática ocasiona.

Conclui-se, portanto, o quanto é importante fazer um estágio antes de entrar no mercado de trabalho, pelo fato de que a transformação da teoria em prática gera como conseqüência um preparo técnico e intelectual. Ou seja, o treinamento de estágio, reduz dúvidas e erros, definindo uma trajetória provavelmente de sucesso profissional.

5. REFERÊNCIAS BIBLIOGRÁFICAS

ABRAHAMSSON, P. *et al. New directions on agile methods: a comparative analisys*. IEEE Proceedings of the 25th International Conference on Software Engineering (ICSE´03) 2003.

ALVAREZ, M. *O que é HTML*. 2004. Disponível em: <http://www.criarweb.com/artigos/7.php>. Acesso em: 17 de Julho de 2014.

BASSI FILHO, D. L. *Experiências com desenvolvimento ágil*. 170 pág. Dissertação (Mestrado) - Universidade de São Paulo, São Paulo, 2008.

BECK, K. *Extreme programming explained: Embrace change*. Addison-Wesley, 2000.

BRIDEE, E. *Play Framework – alta produtividade em Java*. 2011. Disponível em:<http://blog.erkobridee.com/2011/12/05/play-framework-alta-produtividade-em-java/>. Acesso em: 19 de Julho de 2014.

CAELUM. *Persistindo os dados com o Hibernate*, 2014. Disponível em:< http://www.caelum.com.br/apostila-vraptor-hibernate/persistindo-os-dados-com-o-hibernate/>. Acesso em: 20 de Julho de 2014.

CAMPIONE, M; WALRATH, K. *The Java Tutorial: Object-Oriented Programming for the Internet.* [S.l.]: SunSoft Press, 1996.

CHAUBEY, R.; SURESH, J.K. *Integration vs. development: an engineering approach to building web applications*. IEEE Software, feb. 2001, p. 171-181.

CHUMA, E. *Video: Mootools - Curso de Javascript Completo - Aula 47.* s. d. Disponível em <http://www.devmedia.com.br/mootools-curso-de-javascript-completo-aula-47/29954>. Acesso em: 19 de Julho de 2014.

COCKBURN, A.; HIGHSMITH, J*. Agile software development: The human factor*. IEEE Computer, nov. 2001, p. 131-133.

COCKBURN, A. e HIGHSMITH, J. *Agile Software Development: The Business of Innovation*, IEEE Computer, Sept., (2001), pp. 120-122

COCKBURN, A. *Agile software development*. Addison Wesley, The agile software development series, 2002.

CONALLEM, J.: *Building Web Applications with UML*. Addison Wesley, 2000.

FAYAD, M. E., Schimidt & D. C., Johnson, R. E.*Implementing application frameworks: object-oriented frameworks at work*. New York: J. Wiley, 1999, p. 729.

FENTON, N. E; PFLEEGER, S. L. *Software metrics: a rigorous approach*, 2a ed. International Thomson Computer Press, 1997.

FERREIRA, D. *Criando uma aplicação simples com AngularJS*. 2012. Disponível em: <http://tableless.com.br/criando-uma-aplicacao-simples-com-angularjs/>. Acesso em: 18 de Julho de 2014.

FOWLER, M. *The new Methodology*. Disponível em: <http://martinfowler.com/articles/newMethodology.html>.Acessado em 23 jan. 2014.

HIGHSMITH, J. A. *Adaptive Software development: acollaborative approach to managing complex sustems*. DorsetHouse, 2000.

HIGHSMITH, J. A. *History: The agile manifesto*. Disponível em:<http://www.agilemanifesto.org/history.html>. Acessado em 18 jan. 2014.

HIGHSMITH, James A. *Agile software development ecosystems*.Addison-Wesley, The agile software development series, 2002.

JUNIOR, O. *Programação - Apostila Java Script.* s. d. Disponível em: <http://www.ebah.com.br/content/ABAAAfiNYAD/programacao-apostila-java-script>. Acesso em: 20 de Julho de 2014.

MANIFESTO *for agile software development*. Feb. 2001, Disponível em: <http://www.agilemanifesto.org/>. Acessado em 21 jan. 2014.

MACHADO, M.; MEDINA, S. G. *SCRUM – Método Ágil: uma mudança cultural na Gestão de Projetos de Desenvolvimento de Software*, s.d.

MOTA, K. *Primeiros passos no uso da Spring MVC*. 2011. Disponível em:< http://www.klebermota.eti.br/2011/11/21/primeiros-passos-no-uso-do-spring-mvc/>. Acessado em 20 de Julho de 2014.

PALMER, S.; FELSING, J. *A. practical guide to Feature Driven Development*. Prentice-Hall Inc, 2002.

PEREIRA, Ana Paula. *O que é CSS?.* 2009. Disponível em: <http://www.tecmundo.com.br/programacao/2705-o-que-e-css-.htm>. Acesso em: 18 de Julho de 2014.

PEREIRA, Paulo. *et al. Entendendo Scrum para Gerenciar Projetos de Forma Ágil. 2007.*

RAMOS, E. A. *Metodologias Ágeis: Extreme Programming (XP*), 2013.

ROSE, J. *Desenvolvimento Flex com o Swiz Framework*. 2009. Disponível em:<http://www.infoq.com/br/news/2009/02/swiz-framework>. Acesso em: 20 de Julho de 2014.

SCHWABER, K.; BEEDLE, M. *Agile software development with Scrum*. Prentice-Hall, Inc., 2002.

SCHWABER K., *Agile Project Management With Scrum*, Microsoft Press, 2004.

SILVA, M. *JQuery: A Biblioteca do Programador JavaScript*. 2008. Disponível em:<http://livrojquery.com.br/index-1ed.php>. Acesso em: 18 de Julho de 2014.

SOARES, M. *Metodologias Ágeis Extreme Programming e Scrum para o Desenvolvimento de Software*, s.d.

TELES, V. M. *Extreme Programming: Aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade*. Novatec, 2004.