**4 Fundamentação Teórica**

**4.1 Web 1.0 e o surgimento do manifesto ágil.**

A Web 1.0, era uma plataforma, onde usuários, no caso espectadores, podiam apenas navegar entre as páginas e visualizar seu conteúdo; não sendo permitido nenhum tipo de interação com o serviço, como por exemplo, reeditar ou alterar algum conteúdo exposto na página.

        Além disso, a grande maioria de serviços prestados na Internet, eram liberados mediante a pagamentos, ou seja, para usufruir de algum conteúdo, era necessário realizar um pagamento e uma licença única e restrita àquele usuário era gerada para que o mesmo tivesse acesso a tais serviços.

         Apesar do vasto conteúdo de informações transmitidas para o usuário, a Internet era aberta para todos, sem  nenhum dono para controlar os dados acessados.

Eis que surge a segunda fase da Internet que foi o marco da evolução da Internet, onde Tim O’Reilly no ano de 2005 diz as seguintes considerações:

*A web 2.0 é a mudança para uma Internet como plataforma, e um entendimento     das regras para obter sucesso nesta nova plataforma. Entre outras, a regra mais importante é desenvolver aplicativos que aproveitem os efeitos de rede para se tornarem melhores quanto mais são usados pelas pessoas, aproveitando a inteligência coletiva. (REILLY, 2005). [5]*

      Nesta mudança, obteve-se um enorme avanço em relação à interação do usuário, publicações de conteúdo, edições, criações de páginas online sem nenhuma dificuldade em nível de programação e também pode-se citar que o utilizador disponibilizava servidores gratuitamente.

Blogs, PodCast, Hi5, De.li.cio.us são alguns termos que podem ser citados como exemplo da grande variedades de serviços que estão hoje disponíveis graças ao avanço que a Web 2.0 propôs. [6]

       Outras características importantes, são as informações que se atualizam constantemente a fim de diminuir e obter correções para maior comodidade do utilizador, interações com pessoas do mundo inteiro, nisso, incluímos aplicações como Messenger, Googletalk, Skype e Voip, algumas ferramentas de escrita online como Google Docs e Podcast e o avanço do acesso a vídeos que cresceu muito nos últimos anos com o surgimento do Youtube, GoogleVideos e YahooVideos.

Sendo assim, com o número de usuários na web crescendo a todo instante, a demanda de programação para a web foi crescendo cada vez mais, e mesmo com a evolução dos computadores e tecnologias, problemas como, prazos de entrega, construção de softwares confiáveis, custos. [7] Até mesmo o não entendimento por parte da parte dos programadores que ficavam um pouco confusos na hora de ser feita a interpretação dos textos da documentação, surgiu um novo modo de ser feita a organização dessas documentações. O Manifesto Ágil não ignora os processos e ferramentas, documentação, ou a negociação e contratos, somente mostra que tais têm importância secundária num projeto, algo que tornaria a prática na implementação do software em algo muito mais eficiente e com bastante clareza.[8]

Antes do manifesto, vários fatores agravavam o não cumprimento de metas e prazos de entrega dos softwares aos clientes, tais como poucas reuniões com os clientes, onde ficavam faltando informações sobre o projeto, advogados que brigavam por causa de contratos, documentação muito extensa que acarretava o desentendimento dos desenvolvedores na hora de analisar a documentação.

O termo, desenvolver com agilidade ou “Metodologias Ágeis”, teve início no ano de 2001, e para este método foram estabelecidos princípios e características comuns destes métodos. Assim foi criada a “Aliança Ágil” e efetuou-se o estabelecimento do “Manifesto Ágil”.

Extreme Programming (XP) é uma metodologia voltada para pequenas empresas que desenvolvem projetos que contém requisitos básicos, porém que se modificam com uma certa frequência. Entre suas diferenças com as metodologias de desenvolvimento clássica, estão o feedback que tem que ser constante, a abordagem incremental e a boa comunicação entre cliente e desenvolvedor. Tem como seus principais objetivos a agilidade no processo de desenvolvimento e buscando sempre a satisfação do cliente.. Suas práticas são conduzidas por quatro princípios básicos, sendo eles:

1. **Princípio de comunicação** – busca manter o relacionamento entre cliente e desenvolvedor sempre da melhor forma possível, com reuniões práticas e esclarecedoras, prezando sempre pelo contato pessoal.
2. **Princípio da simplicidade** - busca-se alcançar o objetivo do projeto de software com o menor número possível de classes e métodos, também está em sua característica a procura da implementação de requisitos atuais, implementando algo simples do quer realizar a implementação de requisitos complicados que talvez nunca venham a ser usados.
3. **Princípio do feedback** – o desenvolvedor terá constante feedback do código que está sendo construído se informando num geral sobre todo o projeto e recebendo também feedback do cliente para qual o projeto é destinado. A informação do código é conseguida através de constantes testes em que indicam os erros que o código possui, tanto erros pessoais quanto erros de código integrado.
4. **Princípio da coragem** – Nem todas as pessoas tem facilidade de se expressar e se comunicar, sendo assim, o método prezando pela simplicidade, os desenvolvedores podem experimentar e buscar novas soluções para os problemas, além do que, é preciso coragem para cobrar o cliente por constantes feedback. [9]

As práticas do modo XP começam pelo planejamento, sendo definido o que é, e o que não é necessário ser feito no projeto, baseando-se em requisitos atuais, deixando que requisitos futuros sejam implementados somente quando haja a necessidade deles no projeto. A entrega do software acontece conforme os requisitos vão sendo implementados, a cada requisito há uma atualização do software, cada versão entregue ao cliente deve conter o menor tamanho possível, contendo os requisitos de maior valor para o negócio.

A descrição do projeto fica por conta de metáforas, sendo assim, a descrição não conta com termos técnicos, o que possibilita um maior entendimento para o cliente, sendo muito importante a participação do cliente em todo o processo de desenvolvimento para sanar dúvidas de requisitos, evitando atrasos ou até mesmo construções erradas.

A Extreme Programming (XP) prioriza a validação do projeto durante todo o processo de desenvolvimento. Os desenvolvedores implementam o software criando primeiramente os testes, sendo a implementação de código feito em duplas, ou seja, dois desenvolvedores trabalham em um único computador, procurando identificar erros sintáticos e semânticos, pensando em conjunto em como melhorar o código que está sendo implementado.

Muito importante também é a refatoração do código, tornando um código lapidado, limpo, para que não haja problema na hora que for feita a leitura deste código que pertence a todos os membros da equipe. Isso significa que qualquer pessoa pode perceber alguma necessidade no código, pode fazê-lo mesmo sem ter sido a pessoa propriamente dita quem desenvolveu aquele código, mas desde que faça os testes necessários e não prejudique as funcionalidades atuais. A grande vantagem de todos ter acesso a todo o código, é que se algum membro abandona o projeto, outro membro não terá grandes dificuldades ao se encarregar das tarefas que foram deixadas pelo outro membro do projeto, pois todos conhecem todas as partes do software mesmo que não seja de forma detalhada.

A XP também assume, trabalhar muitas horas por dia pode acabar acarretando um mal desempenho pois o programador acaba ficando cansado, que por consequência acaba cometendo mais erros e pra corrigi-los, leva ainda mais tempo, Então diz-se que não se deve trabalhar mais que 40 horas semanais. Caso seja necessário trabalhar mais que 40 horas, significa que existe um problema sério no projeto que deve ser resolvido não com o aumento de horas trabalhadas, mas sim como melhorar o planejamento. [10]

Outro modelo de desenvolvimento ágil é o Scrum [11]. Seu objetivo é manter um processo de desenvolvimento favorável para o projeto de desenvolvimento orientado a objeto. O objetivo desse modelo é manter um ambiente de desenvolvimento flexível em um ambiente de trabalho onde o projeto passa por várias mudanças.

No desenvolvimento de software ágil mudamos nosso processo de gerenciamento do projeto para no quais obtêm software funcional feito um pouco de cada vez. Desenvolvimento ágil derruba diversos paradigmas comparados com o modo de desenvolvimento que era utilizado até então, seguindo seus princípios fundamentais.

Ao se desenvolver um software é muito comum que os requisitos mudem com certa frequência, enquanto ainda a implementação está sendo realizada, mudanças em toda documentação do projeto, levariam muito tempo e tornaria o projeto imprevisível e complexo, pondo em risco a data de entrega ao cliente. Sendo assim, a ideia é a entrega de um produto que é realmente útil para o cliente. [12]

É necessário e essencial nos dias de hoje utilizar o desenvolvimento ágil de software, pois este traz mudanças na forma como o software é desenvolvido, mudando paradigmas do processo de desenvolvimento que vem nos acompanhado por décadas, onde o desenvolvimento tradicional tem fases e checkpoints bem definidos durante seu processo de desenvolvimento onde no final de cada fase algum objetivo é cumprido e algo é entregue.

O desenvolvimento ágil tem iterações ao invés de fases onde no final de cada iteração temos código funcional que pode ser usado e avaliado, podendo responder aos requisitos do usuário que estão sempre mudando e evoluindo.

As chamadas metodologias de desenvolvimento tradicionais, devem ser aplicadas apenas para projetos em que não haverá muitas mudanças no decorrer da implementação, alterações onde a necessidade de refazer o código não for muito grande e não apresente um alto custo.

Além da evolução em como a internet era vista pelos navegadores, após a evolução da web 1.0 para web 2.0 como citado acima, o desenvolvimento também sofreu uma evolução, a maioria dos sites antigos eram todos escritos em HTML.

HTML foi projetado para definir a estrutura de um documento da web, onde o código <p> é um elemento estrutural que se refere à "parágrafo", <LI> também é um elemento estrutural que se refere à "listar item" e enquanto o HTML era expandido, mais elementos eram adicionados. Estes elementos, definem como o texto que será exibido, será formatado.

Em outras palavras, forma e conteúdo se tornaram inseparáveis em HTML, porém o XML foi projetado para fazer exatamente isso onde <title> não define a forma e sim o conteúdo, o mesmo com <link> e <description>, assim os dados podem ser exportados, livres de restrições de formatação.

Livres de restrições e com a popularização da internet, usuários não precisavam mais saber programas para criar um espaço próprio na Internet, nascem então os blogs, onde a programação já era toda definida pelo servidor em que hospedava os blogs, o usuário a partir de sua conta, somente tinha que enviar informações em forma de textos simples, que seria exibida em sua página.

Mesmo sendo um avanço tecnológico grande, para se construir um site, o programador dedicava uma grande parte do seu tempo para digitar inúmeras linhas de comando, as vezes para se construir pouca coisa, necessitava de um grande número de linhas.

Com o aprimoramento das tecnologias, novas linguagens de programação para web foram surgindo, e que satisfazem essa questão da necessidade do programador digitar várias linhas de comando. Exemplo dessas tecnologia de desenvolvimento ágil é a linguagem Ruby que foi escolhida por ser uma linguagem dinâmica e atual, com constante crescimento no mercado de trabalho, segue abaixo suas definições e ferramentas que podem ser usadas juntamente com a linguagem.

**4.2 Ruby**

Ruby é uma linguagem de programação interpretada multiparadigma, de tipagem dinâmica e forte, com gerenciamento de memória automático, originalmente planejada e desenvolvida no Japão em 1995 por Yukihiro “Matz ”Matsumoto, para ser usada como linguagem de script. Matz queria uma linguagem de script que fosse mais poderosa do que Perl e mais orientada a objetos do que Python.

A linguagem também suporta programação funcional, orientada a objetos, imperativa e reflexiva. Foi inspirada principalmente por Python, Perl, Smalltalk, Eiffel, Ada e Lisp, sendo muito similar em vários aspectos a Python. Atualmente, Ruby é a 11ª (décima primeira) linguagem de programação mais popular do mundo, de acordo com o Índice Tiobe [13]

**4.3 Rails**

Rails é seu framework, foi escrito por David Heinemeier Hansson em Ruby, organizado em MVC (Model-View-Controller). Lançado para o público em 2004, sua característica é a facilidade na hora de programar, tornando códigos menores o que leva o programador programar mais em menos tempo.

Este framework surgiu para trazer uma forma mais dinâmica para o desenvolvimento de sites orientado a banco de dados, focando na agilidade para aumentar a produtividade em uma quantidade de linhas menores.

Esta é uma plataforma com opinião, ela impõe que tem um modo certo de se fazer as coisas e foi desenvolvido para encorajar o programador a seguir este modo.

Rails segue algumas filosofias, e uma delas é denominada DRY (Don't Repeat Yourself) ela diz que não é bom que não é bom escrever o mesmo código mais de uma vez.

Em uma plataforma de arquitetura MVC (Model-View-Controller) o ROR utiliza um recurso chamado Scaffolding que gera automaticamente controllers e views a partir de um certo modelo mostrado para o programador. RoR também provê abstração de SGBDs (Sistema de Gerenciamento de Banco de Dados), o usuário usa apenas uma sintaxe para MySQL, PostgreSQL, SQL Server, Oracle, Firebird, entre outros

O Rails faz suposições sobre o que você quer fazer e como você estará fazendo isto, em vez de deixá-lo mudar cada minúscula coisa através de intermináveis arquivos de configuração. [14]

Esta ferramenta poderosa de programação para Web pode ser rodado em Windows, Linux e Mac OS e a permissão para o uso do software é gratuitamente, podendo modificar sem restrição, com direitos a usar, copiar, distribuir e até vender.

**4.3 CoffeScript**

A linguagem disponibiliza o uso do coffescript que é uma pequena linguagem de script que compila para JavaScript para desenvolvimento de websites, é apenas JavaScript, compilando de um-para-um para o js (JavaScript) equivalente. CoffeScript tenta mostrar as boas partes do JavaScript em um modo simplificado. Esta linguagem converte os scripts .coffe em scripts .js (JavaScript). [15] Segue abaixo exemplos de sintaxes escrita em JavaScript e a mesma sintaxe escrita em CoffeScript.

Imagem 1 – Exemplo Coffescript



**4.4 HAML**

Além do CoffeScript, o uso do HAML também é utilizado. A ideia do HAML é somente a estética da sintaxe, não funcionalidades. É uma sintaxe simplificada do HTML, renderizado para HTML que não necessita o fechamento de tags. Segue abaixo exemplo de sintaxe escrita em HTML e a mesma escrita em HAML.

Imagem 2 – Exemplo HAML



**4.5 SASS**

Quando o assunto diz respeito a CSS, a tecnologia utilizada chama-se SASS, que é uma extensão para o CSS3, desenvolvida Nathan Weizenbaum, adicionando regras aninhadas, variáveis, mixins, herança de seletor. É traduzida para o formato CSS padrão. [16]

Imagem 3 – Exemplo SASS



5 O'Reilly, T. (2005) What Is Web 2.0: Design Patterns and Business Models for the Next Generation of Software Disponível em: <http://oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html>. Acesso em 13 julho 2012

6 Richardson, W. (2006). Blogs, Wikis, Podcast and other powerful Web tools for classrooms. Thousand Oaks, CA: Corwin Press.

7 Standish Group, “CHAOS report”, 586 Olde Kings Highway, Dennis,

MA 02638, USA, (1995)

8 Soares, S, M. (2004) Metodologias Ágeis Extreme Programming e Scrum para o desenvolvimento de software. Disponível em: <revistas.facecla.com.br/index.php/reinfo/article/download/146/38> Acesso em 14 de julho 2012

9 Conceitos básicos sobre Metodologias Ágeis para Desenvolvimento de Software (Metodologias Clássicas x Extreme Programming). Disponível em: <http://www.devmedia.com.br/conceitos-basicos-sobre-metodologias-ageis-para-desenvolvimento-de-software-metodologias-classicas-x-extreme-programming/10596> Acesso em 14 de julho 2012

10 BECK, Kent; FOWLER, Martin. (2000) Planning Extreme Programming

11 Schwaber, K. e Beedle, M. "Agile Software Development with SCRUM", Prentice-Hall, (2002)

12 Schwaber, K. "Scrum Development Process", OOPSLA'95 Workshop on Business Object Design and Implementation. Springer-Verlag. (1995)

13 Bruce, S. (2001). An Interview with the Creator of Ruby. Disponível em <http://linuxdevcenter.com/pub/a/linux/2001/11/29/ruby.html>. Acesso em 14 julho 2012

14 Ruby. O que é Ruby on Rails? (2012) Disponível em <http://ruby-br.org/?page\_id=89> Acesso em 26 de Maio 2012.

15 Jeremy, A. CoffeScript (2011). Disponível em <http://coffeescript.org/> Acesso em 16 de julho 2012.

16 Sass. Syntactically Awesome Stylesheets. (2011). Disponível em <http://sass-lang.com/> Acesso em 16 de Julho 2012.