**Serviço Nacional de Aprendizagem Comercial**

**Faculdade Senac Porto Alegre**

**Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas**

**MARCELO BALDISSERA CURE**

**PLANO DE TRABALHO**

**SISTEMA DE COLETA DE MÉTRICAS DE PROJETOS ÁGEIS PARA PROVER ESTIMATIVAS PRECISAS**

Porto Alegre

2014

**MARCELO BALDISSERA CURE**

**PLANO DE TRABALHO**

**SISTEMA DE COLETA DE MÉTRICAS DE PROJETOS ÁGEIS PARA PROVER ESTIMATIVAS PRECISAS**

Plano de Trabalho, apresentado como requisito parcial à obtenção da aprovação do projeto de TCC1 do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas, pela Faculdade Senac Porto Alegre.

Orientador: Prof. Me. Luciano Zanuz

Porto Alegre

2014

**LISTA DE ILUSTRAÇÕES**

**LISTA DE TABELAS**

**LISTA DE SIGLAS**

ADS Análise e Desenvolvimento de Sistemas

SENAC Serviço Nacional de Aprendizagem Comercial

SUMÁRIO

[1 APRESENTAÇÃO GERAL DO PROJETO 8](#__RefHeading__814_242833606)

[2 DEFINIÇÃO DO PROBLEMA 17](#__RefHeading__816_242833606)

[3 OBJETIVOS 18](#__RefHeading__818_242833606)

[3.1 Objetivo Geral 18](#__RefHeading__820_242833606)

[3.2 Objetivos Específicos 18](#__RefHeading__822_242833606)

[4 ANÁLISE DE TECNOLOGIAS/FERRAMENTAS 21](#__RefHeading__824_242833606)

[4.2 Linguagem de programação 21](#__RefHeading__675_295579539)

[4.2 Ferramenta de Desenvolvimento 21](#__RefHeading__626_1177024144)

[4.1 Banco de Dados 21](#__RefHeading__677_295579539)

[4.3 Controle de Atividades 21](#__RefHeading__628_1177024144)

[4.4 Versionamento do Código 22](#__RefHeading__630_1177024144)

[4.5 Qualidade do código 22](#__RefHeading__632_1177024144)

[4.6 Ferramentas de Modelagem 23](#__RefHeading__707_1482487702)

[4.6.1 Protótipos de tela 23](#__RefHeading__709_1482487702)

[4.6.2 Modelo ER 23](#__RefHeading__711_1482487702)

[5 DESCRIÇÃO DA SOLUÇÃO 24](#__RefHeading__826_242833606)

[6 ABORDAGEM DE DESENVOLVIMENTO 25](#__RefHeading__828_242833606)

[7 ARQUITETURA DO SISTEMA 26](#__RefHeading__830_242833606)

[7.1 Modelos UML 26](#__RefHeading__652_47134840)

[7.2 Modelagem de Dados 26](#__RefHeading__654_47134840)

[7.3 Interface Gráfica 26](#__RefHeading__656_47134840)

[8 VALIDAÇÃO 27](#__RefHeading__832_242833606)

[9 CRONOGRAMA 28](#__RefHeading__834_242833606)

[REFERÊNCIAS BIBLIOGRÁFICAS 37](#__RefHeading__836_242833606)

[COMPONENTES REUTILIZADOS 38](#__RefHeading__838_242833606)

# 1 APRESENTAÇÃO GERAL DO PROJETO

No desenvolvimento de software em geral, a estimativa de esforço para implementar um requisito pode afetar o projeto em alguns fatores, como o prazo que pode ser comprometido, podendo acarretar em atrasos de entrega. As estimativas são peça fundamental para o sucesso de um projeto.

Atualmente, as metodologias ágeis estão se tornando cada vez mais populares na indústria do desenvolvimento de software (https://www.planbox.com/agile-by-the-numbers-2013-performance-study/). Estas metodologias tendem a diminuir o risco do projeto fazendo entregas periódicas chamadas de iterações.

Seguindo na linha das metodologias ágeis, uma das mais conhecidas e utilizadas é o Scrum. Esta metodologia prega entregas frequentes de funcionalidades, reuniões diárias para sincronização da equipe, entre outras práticas. Ao final de cada iteração, chamada de sprint, funcionalidades completas são entregues, possibilitando ao cliente prover um feedback e planejar as próximas sprints (METHODOLOGY, 2009)(https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/2013/Scrum-Guide-Portuguese-BR.pdf).

Projetos executados com *Scrum* normalmente tem *sprints* pequenas, onde os requisitos são quebradas em estórias. Cada estória é estimada, normalmente, utilizando o *Planning Poker*, onde os componentes da equipe as estimam cada estória em pontos*.*

A proposta deste sistema é coletar a quantidade de pontos entregues ao final de cada sprint*.* Com estes dados coletados, o sistema poderá prover a quantidade média de pontos que a equipe é capaz de entregar em uma sprint futura.Isto acarreta em mais segurança para o fornecedor e para o cliente. A equipe de desenvolvimento saberá quantos pontos poderão ser entregues e o cliente, juntamente com a equipe, poderá priorizar os requisitos para a sprint evitando atrasos.

# 2 DEFINIÇÃO DO PROBLEMA

Quando os requistos de um projeto são definidos e quebrados em tarefas menores, estes necessitam de uma estimativa de esforço para executá-los. A estimativa de esforço é um fator importante para o sucesso ou fracasso de um projeto, pois ela está diretamente ligada com o prazo e o custo. De acordo com estatísticas, apenas 35% dos projetos tiveram sucesso em 2006 (https://www.ibm.com/developerworks/community/blogs/tlcbr/entry/fim\_do\_empirismo?lang=en)

Se o time de desenvolvimento de software pudesse prover estimativas de esforço precisas, este seria um cenário ideal tanto para o cliente quanto para o time de desenvolvimento. Os projetos poderiam ser vendidos com mais segurança para o cliente e para o time, estipulando o prazo e o custo sem surpresas ao final do projeto.

Projetos executados com metodologias ágeis, como o Scrum, tendem a ter entregas periódicas de funcionalidades específicas acordadas entre o cliente e o time de desenvolvimento, conforme ilustrado na Figura 1. Estes ciclos são chamados de Sprints. Normalmente são períodos curtos, como por exemplo, duas semanas. Ao final de cada Sprint as funcionalidades são entregues ao cliente, e uma reunião entre cliente e equipe é feita para definir as estórias que deverão ser entregues no próximo ciclo.

Figura 1: Processo do Scrum (http://epf.eclipse.org/wikis/scrumpt/Scrum/guidances/supportingmaterials/resources/ScrumLargeLabelled.png)

Estórias são os itens que vao ficar, inicialmente, no backlog do projeto. O time escreve as estórias baseando-se no ponto de vista do cliente. Pode-se considerar que a estória é um requisito do sistema. Por exemplo, ao desenvolver uma calculadora, podemos definir as estórias como: subtração, adição, etc (http://scrummethodology.com/scrum-user-stories/).

As estórias priorizadas para serem executadas na Sprint devem ser estimadas. Normalmente as estimativas são feitas utilizando a metodologia Planning Poker, onde cada estória é estimada em pontos, que variam de 0 até 100 pontos (<http://www.planningpoker.com/>).

O Planning Poker basicamente é uma metodologia utilizada especificamente para o time fazer estimativas de esforço. Depois que a estória é apresentada para os membros do time, este discute a estória para ter certeza de que todos entenderam. Cada membro do time escolhe uma pontuação representando o esforço que será necessário para implementar estória, e não mostra para ninguém. Depois que cada membro do time escolheu uma pontuação, chamada de Story Point, os membros do time voltam a discutir para chegar a um denominador comum, então o time passa a ter a estimativa, em Story Points, para aquela estória específica.

É comum que a pontuação estimada no Planning Poker para a estória não seja precisa, por exemplo, o time estimou a estória em 5 pontos, mas esta poderia ser estimada em 3 pontos, pois o esforço gasto foi proporcional a uma estória de 3 pontos.

Este trabalho visa facilitar as estimativas de estórias baseado em dados históricos de Sprints passadas. Ao final de cada Sprint o sistema deverá coletar quantos pontos a equipe prometeu entregar ao final da Sprint e quantos pontos realmente foram entregues. Então baseando-se nas Sprints passadas, as estimativas se tornam cada vez mais precisas e as Sprints podem ser planejadas com mais segurança e precisão. Tudo isto agrega valor para o negócio do cliente.

# 3 OBJETIVOS

Este capitulo descreve os objetivos gerais e específicos do projeto.

## 3.1 Objetivo Geral

O objetivo geral deste trabalho é facilitar as estimativas de estória, provendo segurança e precisão ao time e ao cliente ao planejar a sprint. O sistema se baseará em dados históricos de sprints passadas para prover estatísticas úteis para facilitar o planejamento da sprint.

## 3.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

* Prover confiança para o time de desenvolvimento e para o cliente ao planejar a Sprint;
* Garantir a entrega da sprint;
* Garantir, de forma geral, a entrega de uma release;
* Integrar o sistema com o software Trello ([http://www.trello.com](http://www.trello.com/));
* Coletar informações pertinentes dos boards do trello;
* Gerar relatórios para auxiliar nas estimativas futuras;'

# 4 ANÁLISE DE TECNOLOGIAS/FERRAMENTAS

Este capitulo descreve as tecnologias que serão utilizadas para o desenvolvimento do projeto, incluindo tecnologias de desenvolvimento, modelagem e técnicas.

# 4.2 Linguagem de programação

Este projeto será desenvolvido utilizando a linguagem de programação Python, pois ela provém facilidade no desenvolvimento, as APIs são simples e bem documentadas, assim como a própria linguagem. É fácil de configurar bibliotecas externas. Python é uma linguagem de programação grátis, onde não é necessário pagar licença. (<http://www.python.org.br/>).

Será utilizado o framework Django para auxiliar no desenvolvimento das páginas e integração com banco de dados, onde o desenvolvedor pode definir o modelo de dados utilizando a linguagem Python. Este framework também provém uma API poderosa de integração com o banco de dados. Django é um framework para desenvolvimento web com Python e serve, principalmente, para agilizar o desenvolvimento. (<https://www.djangoproject.com/>).

# 4.2 Ferramenta de Desenvolvimento

O desenvolvimento será feito utilizando uma ferramenta chamada Sublime. Esta ferramenta facilita o desenvolvimento pois ela oferece muitas funcionalidades úteis como auto complete, lista de arquivos do projeto, multi seleção, entre outras. O Sublime é uma ferramenta extremamente leve. ([http://www.sublimetext.com](http://www.sublimetext.com/)).

# 4.1 Banco de Dados

Os dados serão guardados no banco de dados PostgreSQL por ser poderoso e grátis. O PostgreSQL provém a capacidade de escrever consultas complexas e retornar os dados em um formato específico para que o sistema possa utilizá-los de maneira mais facil (<http://www.postgresql.org/>).

O PostgreSQL é um banco de dados relacional open source de fácil instalação de manutenção. Ele pode ser instalado na maioria dos sistemas operacionais.

# 4.3 Controle de Atividades

Para a parte de controle de atividades e sprints, será utilizado o software Trello ([http://www.trello.com](http://www.trello.com/)), o qual é usado como um Kanban. O Trello é um software grátis que possibilita a criação de lists, cards. Além disto é possível também adicionar anexos aos cards e outras funcionalidades interessantes. Este possibilitará fazer o controle das sprints de forma organizada e visual, e as tarefas de sprints passadas podem ser arquivadas de forma a manter um histórico.

# 4.4 Versionamento do Código

Para controle de versão de código e da própria documentação do projeto, será utilizada a ferramenta GitHub ([http://www.github.com](http://www.github.com/)). É um software grátis, onde, registrando um usuário, este pode cadastrar vários projetos, possibilitando a criação de bugs, melhorias, outros usuarios podem contribuir para o projeto, etc. O GitHub será importante para o projeto na parte de versionamento de código, onde tudo será salvo na nuvem e não tem perigo de perder dados em caso de incidentes imprevistos.

# 4.5 Qualidade do código

Falando de qualidade de código, o sistema será implementado utilizando um método de desenvolvimento chamado TDD (Test Driven Development), que basicamente consiste em desenvolver um sistema baseando-se em testes. Antes de desenvolver uma funcionalidade, escreve-se testes para esta. Estes testes, inicialmente, estarão falhando, pois as funcionalidades ainda não foram implementadas. O primeiro passo seria fazer estes testes passar, implementando as funcionalidades de acordo com tal necessidade. Assim o desenvolvedor tem uma segurança em saber que aquela funcionalidade já possui testes unitários, provendo segurança também para refatorar o código. A qualidade do código pode ser medida pela cobertura de testes para cada módulo ou classe do sistema. O processo básico do TDD está descrito na Figura 2. (<http://www.agiledata.org/essays/tdd.html>).



Figura 2: Estrutura básica do TDD (http://www.agiledata.org/images/tddSteps.jpg)

# 4.6 Ferramentas de Modelagem

Abaixo estão relacionadas as ferramentas utilizadas para fazer as modelagens necessárias para o projeto.

# 4.6.1 Protótipos de tela

Para a criação dos protótipos de tela será utilizado o NinjaMock. Esta ferramenta possibilita criar protótipos de telas de várias plataformas, como Web, Android, iOS, WindowsPhone, etc. O NinjaMock é grátis e possibilita que as imagens possam ser exportadas em PDF ou PNG (<http://ninjamock.com/>).

# 4.6.2 Modelo ER

Para a criação do modelo ER, será utilizado o Cacoo. Esta é uma ferramenta que possibilita a criação de diversos tipos de diagramas, como por exemplo diagrama ER e UML. Cacoo é uma ferramenta gratis. (<https://cacoo.com/>).

# 5 DESCRIÇÃO DA SOLUÇÃO

O sistema será restringido somente para pessoas que possuem acesso. Diferentes usuários poderão acessar diferentes projetos, onde terá a opção para fazer a coleta de dados e visualizar relatórios. O sistema incluirá a área do administrador, onde poderão ser criados projetos e adicionar pessoas a estes projetos.

Considera-se que os projetos tenham acompanhamento na ferramenta online Trello. Esta ferramenta disponibiliza boards onde o usuário pode criar listas, cards, adicionar informações, anexar arquivos, etc (http://www.trello.com).

A coleta de métricas basicamente irá acessar a API do Trello em Python. O usuário selecionará um projeto e iniciará a coleta. O sistema irá buscar os cards que estiverem na última coluna do board, que é a coluna “done”, onde estarão as tarefas que foram concluídas na Sprint.

O sistema irá coletar o identificador de cada card, quanto tempo levou para ser finalizado, os pontos, linguagem de programação e número de testes unitários criados para aquela tarefa. A última etapa da coleta de dados é salvá-los no banco de dados PostgreSQL. Cada execução referente a um projeto será considerada uma Sprint.

Serão disponibilizados relatórios de métricas referentes a um projeto e relatórios de métricas referentes a todos os projetos da empresa. Diferentes visões serão extraídas destes dados, assim como quantidade de testes unitários criados por sprint, quantidade de testes unitários criados por card, quantidade de estórias por pontos e quantidade média de pontos por Sprint. A estrutura básica do sistema está detalhada na Figura 3.

Figura 3: Estrutura básica do sistema

# 6 ABORDAGEM DE DESENVOLVIMENTO

O projeto será desenvolvido basicamente utilizando alguma partes do Scrum e a utilização do Kanban. O Scrum é uma metodologia ágil de desenvolvimento de software que normalmente é utilizada em times pequenos, então algumas partes dele passam a ser relevantes para o cenário do projeto.

Utilizando o Kanban, será possível ter, de forma visual, um acompanhamento preciso do progresso da sprint corrente. O Kanban possibilita a identificação de gargalos no desenvolvimento, e através de algumas técnicas no gerenciamento do quadro, ele possibilita a melhoria no fluxo das atividades.

Os ciclos, conhecidos como sprints, serão utilizados priorizando itens do backlog para entregas periódicas. Serão executadas sprints de três semanas, onde ao final de cada sprint as funcionalidades priorizadas serão entregues.

O projeto basicamente será dividido nas etapas descritas abaixo:

* Estrutura básica do projeto com Django;
* Criação de telas principais;
* Integração com a API do Trello;
* Criação do modelo de dados;
* Processo de coleta de dados;
* Geração de relatórios;

# 6.1 Artefatos

# A seguir, segue uma descrição dos artefatos gerados pelo projeto.

# 6.1.1 Product Backlog

# O Product Backlog é uma lista, ordenada por prioridade, de tudo que é necessário para a construção do produto. O Product Backlog nunca está completo, ele é dinâmico e está mudando constantemente identificando o que o produto precisa para ser completo e apropriado.

# 6.1.2 Sprint Backlog

# O Spring Backlog é uma lista de itens do Product Backlog priorizados para serem finalizados na Sprint corrente. Eles são coletados de forma ordenada indicando a prioridade de cada ítem.

# 6.1.4 User Stories (estórias)

# Estórias, baseiam-se no ponto de vista do cliente. Pode-se considerar que a estória é um requisito do sistema. As estórias, basicamente podem ser definidas pelo time, baseando-se na perspectiva de quem vai utilizar a funcionalidade gerada por esta estória. Por exemplo, ao desenvolver uma calculadora, podemos definir as estórias como: subtração, adição, etc. (http://scrummethodology.com/scrum-user-stories/).

# 7 ARQUITETURA DO SISTEMA

A arquitetura do sistema será dividida em Modelos UML, Modelagem de Dados e Interface Gráfica.

# 7.1 Modelos UML

Na modelagem UML, serão gerados os diagramas de Classes e diagrama de casos de uso.

# 7.2 Modelagem de Dados

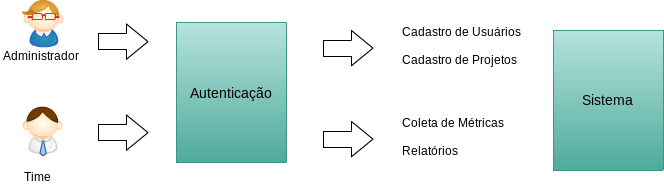
Diagrama Entidade Relacionamento será gerado para demonstrar a modelagem de dados do sistema.

# 7.3 Interface Gráfica

Serão criados protótipos de telas do sistema para ter, de uma maneira visual, uma idéia de como as telas serão construídas. Desta maneira, será possível fazer uma análise de usabilidade para deixar a navegação/execução das tarefas clara e objetiva.

# 7.4 Processo de Negócio

O processo de negócio está descrito na figura 4.

Figura 4 – Processo de Negócio

# 8 VALIDAÇÃO

A validação do projeto será dada através da implantação do sistema na empresa de desenvolvimento de software ilegra ([http://www.ilegra.com](http://www.ilegra.com/)). Este poderá ser utilizada por um projeto específico pra mostrar se realmente foram registradas melhoras nas estimativas e planejamento das sprints.

Serão disponibilizados questionários aos membros do time, onde poderão ser preenchidos após 4 sprints, quando o sistema terá dados históricos suficientes para prover estimativas precisas.

# 9 CRONOGRAMA

Tabela 0 - Cronograma

|  |  |  |  |
| --- | --- | --- | --- |
| Descrição da Atividade | Produto | Data | Descrição |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# REFERÊNCIAS BIBLIOGRÁFICAS

METHODOLOGY, scrum: **Scrum Effort Estimation and Story Points**. 2009. Disponível em: <<http://scrummethodology.com/scrum-effort-estimation-and-story-points/>>. Acesso em: 18 mar. 2014. **(POSTAGEM RETIRADA DE SITE)**

# COMPONENTES REUTILIZADOS