FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Agile *Data Mining*: Uma metodologia ágil para o desenvolvimento de projetos de *data mining*

Diogo Nogueira

Mestrado Integrado em Engenharia Informática e Computação

Orientador: Carlos Soares (PhD)

Coorientadora: Ana Cristina Barros (PhD)

Junho de 2014

Agile *Data Mining*: Uma metodologia ágil para o desenvolvimento de projetos de *data mining*

Diogo Nogueira

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Eugénio Oliveira (PhD)

Vogal Externo: Paulo Cortez (PhD)

Orientador: Carlos Soares (PhD)

_____

23 de Junho de 2014

Resumo

Nos últimos anos tem havido um enorme crescimento e consolidação do campo de *data mining*. Alguns esforços têm sido feitos para procurar estabelecer metodologias para o desenvolvimento de projetos nesta área, tal como o CRISP-DM e o SEMMA. Ainda assim o aumento do número e complexidade dos projetos, a volatilidade dos requisitos e as constantes alterações nos mercados, levam a que se encontrem problemas na metodologias atuais de desenvolvimento.

O principal problema destas metodologias são os ciclos longos até serem apresentados os primeiros resultados. Outro problema importante é a dificuldade na colaboração entre as equipas e o negócio, levando a um desalinhamento de objetivos, a problemas nas dinâmicas de poder, e na alocação de recursos.

Estes são problemas frequentemente encontrados noutra área, a engenharia de software, quando aplicadas metodologias tradicionais para o desenvolvimento dos projetos. Nesta área, as metodologias ágeis têm sido utilizadas com grande sucesso de forma a resolver esses problemas.

Neste projeto é proposta uma metodologia ágil para o desenvolvimento de projetos de *data mining*. A metodologia proposta tem por base a metodologia ágil Scrum, para a gestão do trabalho, e o CRISP-DM como guia para o desenvolvimento de projetos de *data mining*. De forma a verificar a aplicabilidade da metodologia, inicialmente foi testada retrospetivamente em dois projetos, mais tarde de forma a confirmar a sua aplicabilidade em projetos de data mining, foi utilizada num projeto com a duração de cinco semanas e numa competição de *data mining* do *Data Intelligence Group* da FEUP.

Abstract

In the recent years there has been a tremendous growth and consolidation in the field of *data mining*. Some efforts have been made to establish methodologies for the development of projects in this field, such as CRISP-DM and SEMMA. Yet the increasing number and complexity of projects, the volatility of the requirements and the constant changes in the markets, are leading to problems in the current methodologies.

The main problems found with the use of these methodologies are the long cycles before being presented the first results. Not facilitating the collaboration between teams and the business, leading to misalignment of objectives, problems in the power dynamics and problems in resources allocation.

Most of these problems are also found in another field, software engineering, when applied traditional methodologies for project development. Is this field, agile methodologies have been used with great success to solve these problems.

This report describes the state of the art in the fields of *data mining*, agile methodologies, and a proposal of an agile methodology for the development of *data mining* projects.

The proposed methodology is based on Scrum as an agile methodology for work management, and CRISP-DM as a guide for the development of *data mining* projects. In order to verify the applicability of the methodology, it was initially tested retrospectively in three projects, and in order to confirm it's applicability in data mining projects, it was used in a project with the duration of five weeks and in a *data mining* competition of the *Data Intelligence Group* from FEUP.

Agradecimentos

Quero agradecer em primeiro lugar ao meu orientador, Carlos Soares e à minha coorientadora Ana Cristina Barros, pelo apoio e disponibilidade com que sempre me presentearam. Agradeço também aos elementos das equipas que utilizaram a metodologia proposta e a todos os entrevistados durante a elaboração deste estudo, André Fernandes, Maria Pedroto, Artur Aiguzhinov, Ana Isabel Marques, Pedro Abreu, João Bastos, Joaquim Pratas e Fábio Pinto.

Deixo também um agradecimento especial para a Paula Gomes pelo apoio prestado durante o desenvolvimento deste projeto, e ao professor Ademar Aguiar pela disponibilidade que apresentou para me ajudar no desenvolvimento da metodologia.

Obrigado também a todos os meus professores do MIEIC e colegas com quem aprendi muito.

Por fim, agradeço aos meus amigos e familiares por estarem sempre presentes para me apoiarem quando necessário. Deste grupo de pessoas, destaco a minha namorada Joana Cunha, os meus pais Isabel Pinto e Raul Nogueira, os meus dois grandes amigos João Magalhães e Hugo Rocha e os meus amigos e colegas mais próximos do MIEIC.

Diogo Nogueira

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas e Símbolos

CRISP-DM	Cross Industry Standard Process for *Data mining*
SEMMA	Sample, Explore, Modify, Model, Assess
SAS	Statistical Analysis Systems
XP	Extreme Programming
CRM	Customer Relationship Management
DSDM	Dynamic Systems Development Methodology
MoSCoW	Must, Should, Could, Wont
DIG	Data Intelligence Group
DMS	*Data mining Story*

Capítulo 1

Introdução

O *data mining* é, de uma forma muito simples, definido pela procura de relações, padrões e tendências escondidas em conjuntos de dados de grandes dimensões, sendo caracterizado pela capacidade para lidar com o aumento de dados de negócio e aceleradas alterações no mercado (Han e Kamber, 2006). Estas características ajudam a criar ferramentas com grande poder para quem necessita de tomar decisões. Muitos dos processos de negócio atuais envolvem alguma forma de *data mining* (Alnoukari et al., 2008). Gestão da relação com o cliente, optimização da cadeia de fornecimento, previsão da procura, optimização do *stock* e gestão do conhecimento são alguns exemplos de funções do negócio que têm sofrido um grande impacto das técnicas de *data mining*.

Ainda assim, quem desenvolve projetos de *data mining* enfrenta problemas para determinar que metodologia ou processo deve utilizar para desenvolver o projeto (Ghani e Soares, 2006). Como os requisitos do negócio se tornam cada vez mais dinâmicos e incertos, as abordagens tradicionais, estáticas e pesadas, podem não ser capazes de lidar com eles. Estes problemas não são muito diferentes dos problemas que existem em engenharia de *software*, nomeadamente nas abordagens tradicionais como a abordagem *Waterfall* (Leau et al. 2012). Estas abordagens são baseadas numa série de passos sequenciais como definição dos requisitos, desenvolvimento da solução, teste e manutenção. As metodologias tradicionais de desenvolvimento de *software* necessitam de documentação estável e da definição dos requisitos no início do projeto (Leau et al. 2012), partilhando com as metodologias atuais de desenvolvimento de *data mining* estas suas características. Na área da engenharia de *software*, as metodologias ágeis têm sido propostas para resolver esses problemas.

Os processos ágeis são tipicamente caracterizados pela sua flexibilidade, capacidade de adaptação e partilha de conhecimento, permitindo uma abordagem iterativa, evolutiva e incremental ao desenvolvimento, uma entrega rápida do produto, priorização dos requisitos,

envolvimento ativo dos clientes, redução do custo e do tempo, melhoria na qualidade do software e o aumento da probabilidade de sucesso de um projeto (Keith, 2012).

1.1 Problema

O aumento do número e complexidade dos projetos de *data mining* levam a que se encontrem problemas nas metodologias usadas atualmente pelas equipas de desenvolvimento. Estas metodologias apresentam falta de flexibilidade para lidar com os novos projetos que surgem no *data mining* e falta de suporte às atividades de gestão dos projetos.

As metodologias atuais não facilitam a colaboração entre as equipas de desenvolvimento e o negócio (Alnoukari et al., 2008), levando a um desalinhamento de objetivos, a problemas nas dinâmicas de poder, e problemas na alocação de recursos. Nestas metodologias os requisitos para o sistema são também definidos numa fase muito inicial do projeto, não apresentando mecanismos capazes para lidar com a sua mudança. Este facto conjugado com o de os resultados aparecerem muito tarde no processo, leva a que o resultado final possa não estar de encontro com os objetivos e requisitos do negócio.

1.2 Motivação e Objectivos

Com o objetivo de eliminar os problemas descritos no ponto anterior, nesta dissertação é proposta uma nova metodologia de desenvolvimento de projetos de *data mining*, que transpõe as ideias em torno das metodologias ágeis, desenvolvidas no âmbito da engenharia de software, para as metodologias de desenvolvimento de projetos de *data mining*.

Com o objetivo de verificar a aplicabilidade da metodologia, a mesma foi inicialmente testada retrospetivamente em dois casos, tendo sido num segundo momento aplicada num projeto de *data mining* a decorrer no INESC-TEC, com a duração de quatro semanas, e numa competição de *data mining* do *Data Intelligence Group* da FEUP, com a duração de três semanas.

1.3 Estrutura da Dissertação

Para além da introdução, esta dissertação contém mais 3 capítulos. No capítulo 2, é descrito o estado da arte. No capítulo 3, a metodologia de investigação utilizada e a metodologia desenvolvida. No capítulo 4, os estudos de caso realizados. No capítulo 5 as conclusões e trabalho futuro.

Capítulo 2

Revisão Bibliográfica

Neste capítulo é descrito o estado da arte tendo como foco o *data mining* na Secção 2.1. São também desenvolvidas as metodologias de desenvolvimento de projetos predominantes no *data mining* na Secção 2.2. As metodologias de desenvolvimento de *software* ágeis são descritas e comparadas na Secção 2.3. A Secção 2.4 apresenta as conclusões da revisão bibliográfica.

2.1 *Data mining*

O conceito de *data mining* está ligado ao processo KDD, *Knowledge Discovery in Databases*. Este processo é dividido em seis fases: seleção dos dados, pré processamento, transformação dos dados, *data mining* e avaliação/interpretação dos resultados.

De uma forma simples, *data mining* refere-se à extração de conhecimento, por aplicação de algoritmos específicos, a partir de dados. Consiste no processo de descoberta de conhecimento interessante a partir de dados guardados em bases de dados, armazéns de dados e outros repositórios de informação (Han e Kamber, 2006).

O *data mining* envolve a integração de técnicas de diversas disciplinas, tais como tecnologia de bases de dados e armazém de dados, estatística, computação de alta performance, reconhecimento de padrões, aprendizagem automática, visualização de dados, recolha de informação, processamento de imagem e sinal, e análise temporal e espacial de dados.

As tarefas do *data mining* são bastante diversas e distintas pois podem existir diferentes tipos de padrões numa grande base de dados. São por isso necessários diferentes métodos e técnicas de forma a poder encontrar os diferentes padrões. Baseado no tipo de padrões que se pretende encontrar, as principais tarefas em *data mining* podem ser classificadas como classificação, regressão, associação e agrupamento (Fu, 2008).

A **classificação** é uma das tarefas de *data mining* mais comum. Consiste em examinar certas características nos dados e atribuir uma classe previamente definida. Nesta tarefa, o modelo analisa o conjunto de registos fornecidos, em que cada registo contem uma indicação da classe a que pertence, com o objetivo identificar padrões que possam ser usados para classificar uma nova amostra. Por exemplo, partindo de um conjunto de dados históricos, contendo informação sobre o perfil dos colaboradores de uma empresa: perfil negocial, perfil técnico e perfil negocial. O modelo gerado pela classificação, analisa um novo colaborador e é capaz de identificar em qual categoria se insere.

A **regressão** é similar à classificação, porém é utilizada quando a variável objetivo (dependente) é numérica e não uma categoria. Desta forma, pode-se estimar o valor de um novo registo analisando-se os dados anteriores. Por exemplo, partindo dos valores mensais gastos por diversos tipos de consumidores e de acordo com os hábitos de cada um, o modelo gerado pela regressão é capaz de prever qual o valor que será gasto por um novo consumidor.

A **associação** tem como objetivo encontrar elementos que estejam tipicamente associados numa mesma transação, ou seja, encontrar padrões ou relacionamentos frequentes entre conjuntos de dados. Um exemplo são as regras de associação que têm a forma: Se X então Y. É uma das tarefas mais conhecidas devido aos bons resultados obtidos, principalmente na análise de "cestos de compras", isto é, compra de produtos que estejam associadas entre si.

O **agrupamento** (*clustering*), tem como objetivo organizar um conjunto de objetos em subconjuntos (*clusters*), em que os elementos de cada subconjunto são semelhantes entre si, de acordo com uma, ou mais, características interessantes. Por outro lado devem ser diferentes dos registos dos demais *clusters*. Esta tarefa difere da classificação pois não necessita que os registos sejam previamente categorizados. Isto é, não tem como objetivo classificar, estimar ou prever o valor de uma variável, mas apenas identificar os grupos de elementos similares. As tarefas de agrupamento são utilizadas, por exemplo, para segmentar clientes de acordo com a frequência de aquisição nos diferentes departamentos de uma loja, de forma a selecionar alvos para uma campanha promocional.

2.2 Metodologias de *Data mining*

De seguida são descritas duas metodologias para projetos de *data mining*. O CRISP-DM é uma metodologia genérica que consiste numa sequência de passos usualmente envolvidos num projeto de *data mining*. A outra, o SEMMA, é uma metodologia especifica para a ferramenta SAS Enterprise Miner.

Embora não sejam necessárias todas as fases de cada metodologia para todos os projetos, estas metodologias, estabelecem uma base para a estratégia a adotar para o correto desenvolvimento de um projeto de *data mining*.

2.2.1 CRISP-DM

A metodologia CRISP-DM fornece uma visão geral do ciclo de vida de um projeto de *data mining*. Contém as fases do projeto, as tarefas respetivas e os seus *outputs*. O ciclo de vida de um projeto de *data mining*, nesta metodologia, é decomposto em seis fases que são ilustradas na Figura 1 (Chapman et al., 2010). As setas indicam as dependências mais importantes e frequentes entre as fases. Dependendo do projeto e dos resultados obtidos numa determinada fase, pode ser necessário retroceder para uma fase anterior. Assumindo que o projeto não é cancelado durante a implementação, o processo não é terminado até que uma solução chegue à última fase do ciclo.

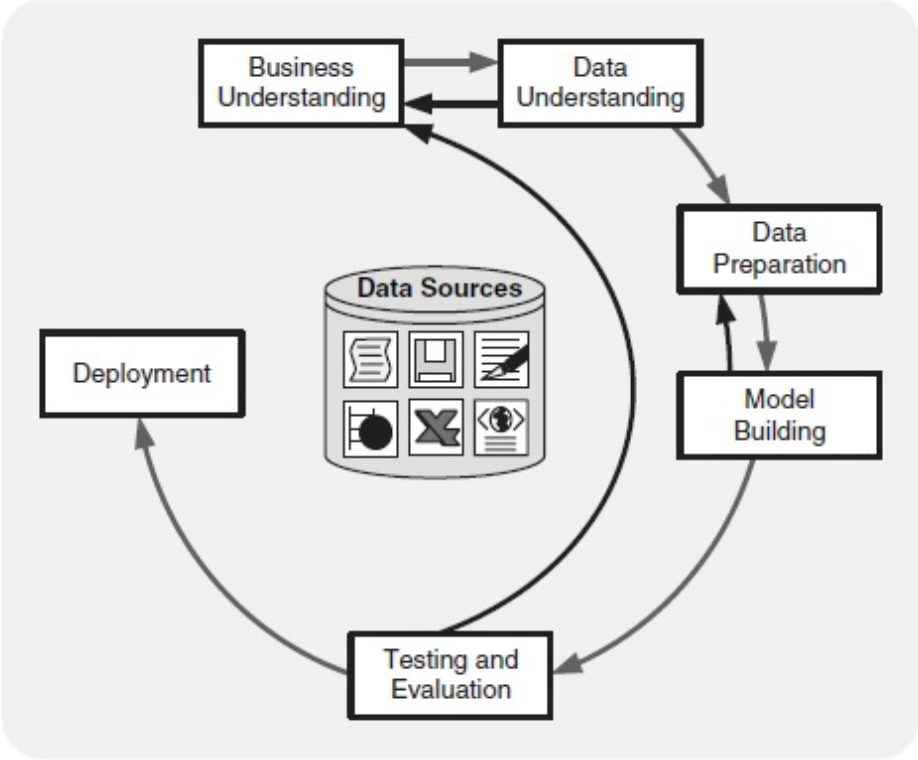

Figura 1: Fases do CRISP-DM (SPSS, 2000. "*Phases of CRISP-DM reference model*". CRISP-DM 1.0, 10)

As seis fases da metodologia CRISP-DM são *Business Understanding, Data Understanding, Data Preparation, Modelling, Evaluation e Deployment* (CRISP-DM 1.0, 2000).

A fase de ***Business Understanding*** tem como objetivo entender os objetivos do projeto e os requisitos de uma perspetiva do negócio. O conhecimento obtido é convertido na definição de um problema de *data mining*, sendo também determinado um plano preliminar para o projeto de forma a atingir os objetivos.

Depois dos objetivos do negócio e do plano do projeto estarem estabelecidos, a fase de ***Data Understanding*** trata das necessidades ao nível dos dados para o projeto. Esta fase pode incluir recolha inicial de dados, descrição dos dados, exploração dos dados e verificação da sua qualidade. Tarefas de *data mining* como *clustering*, podem ser usadas durante esta fase, com o objetivo de perceber melhor os dados. Existe uma ligação próxima entre a fase de análise do problema e a de análise dos dados pois para a formulação detalhada do problema e do plano para o projeto é necessário conhecimento acerca dos dados disponíveis.

Na fase de ***Data Preparation*** são realizadas todas as atividades para a construção do conjunto de dados final, isto é, os dados que serão utilizados nas ferramentas de modelação. As atividades de preparação dos dados, podem ser executadas múltiplas vezes. As tarefas desta fase incluem: seleção de Tabelas, linhas e atributos, limpeza de dados, construção de novos atributos e alteração dos dados para as ferramentas de modelação.

Na fase de ***Modelling*** são selecionadas e aplicadas diversas técnicas de modelação, e os seus parâmetros calibrados de forma a melhorar os modelos obtidos. Nesta fase, é necessário ser realizada uma divisão dos dados em duas partições, uma de treino, a partir da qual serão gerados os modelos, e outra de teste, que será necessária na fase de avaliação dos modelos. Existe uma ligação próxima entre a fase de preparação dos dados e a fase de modelação.

Na fase de ***Evaluation*** já foram construídos um ou mais modelos que parecem ter bastante qualidade, numa perspetiva de análise de dados. É necessário nesta fase avaliar os modelos tendo em conta os objetivos de negócio definidos na primeira fase (análise do problema), de forma a garantir que atinge os objetivos. Um objetivo fundamental é determinar se existe algum requisito do negócio que não foi suficientemente considerado. Esta fase pode levar à identificação de novas necessidades do negócio, muitas vezes pelo reconhecimento de novos padrões nos dados, o que leva ao retrocedimento do processo para fases anteriores do CRISP-DM.

A fase final do ciclo de desenvolvimento do CRISP-DM é a fase de ***Deployment*** dos modelos criados para os processos do negócio. Geralmente, o conhecimento recolhido durante o processo, será organizado e apresentado de uma forma que o cliente o possa utilizar.

Os modelos integrados necessitam de ser monitorizados devido a possíveis alterações nas condições de operação. Se ocorrerem alterações significativas, os modelos devem ser refeitos. É também aconselhável documentar os resultados do projeto de forma a estarem disponíveis para consulta futura.

2.2.2 SEMMA

Para além do CRISP-DM, existe uma outra metodologia para o desenvolvimento de projetos de *data mining*, chamada SEMMA. O nome desta metodologia é o acrónimo correspondente às cinco fases do processo (*Sample, Explore, Modify, Model, Assess*), ilustradas na Figura 2.

O SEMMA consiste num processo para aplicar as ferramentas do SAS Enterprise Miner às tarefas fundamentais de um projeto de *data mining*. Oferece um processo fácil de perceber, permitindo organizar e adequar o desenvolvimento e manutenção de projetos de *data mining*. Apresenta uma estrutura para a concepção, criação e evolução, ajudando a apresentar uma solução para problemas do negócio assim como encontrar as objetivos do negócio para o *data mining*.

Partindo de uma amostra representativa dos dados, o SEMMA tem como objetivo facilitar a aplicação de técnicas de visualização e exploração estatística, selecionar e transformar as variáveis mais significativas, modelar as variáveis de forma a prever resultados, e por fim confirmar a precisão do modelo.

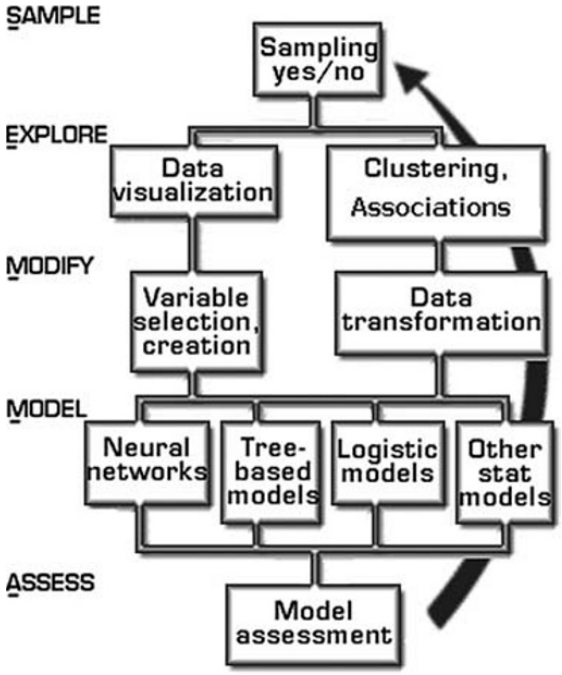

Figura 2: Ilustração do processo SEMMA (D. L. Olson, D. Delen, 2008. "Schematic of SEMMA". Advanced *Data mining* Techniques, 19. Softcover)

A primeira fase do SEMMA é a **amostragem**, onde uma porção dos dados iniciais, grande o suficiente para conter informação relevante mas pequena o suficiente para ser manipulada rapidamente, é extraída. No caso de conjuntos de dados de grandes dimensões, utilizar uma amostra representativa dos dados, em vez do volume total, pode reduzir drasticamente o tempo de processamento necessário para recolher informação crítica para o negócio. Se padrões gerais aparecerem no volume total dos dados, os mesmos serão também detetados numa amostra representativa dos mesmos dados. Caso um padrão seja tão pequeno que não é representado numa amostra e ainda assim tão importante que influencie o modelo, deve ser descoberto utilizando métodos exploratórios descritivos de dados. É também recomendado que sejam

criadas partições de dados, para treino e para validação dos modelos, de forma a aumentar a precisão da análise dos modelos.

A segunda fase do SEMMA é a **exploração**. Nesta fase é realizada uma pesquisa por padrões, tendências e anomalias, de forma a obter uma melhor perceção do conjunto de dados. Depois da fase de amostragem, o passo seguinte é explorá-los visualmente ou numericamente por tendências ou grupos. A exploração ajuda a refinar e redirecionar o processo de descoberta. Se a exploração visual não revelar tendências claras, pode-se explorar os dados a partir de técnicas estatísticas como *clustering* ou *factor analysis*. Por exemplo, num projeto de *data mining* para uma campanha de publicidade por correio, realizar *clustering* pode revelar grupos de clientes com padrões distintos de compras. Limitar o processo de descoberta a cada um dos grupos descoberto, individualmente, pode aumentar a probabilidade de encontrar um padrão mais rico que podia não ser forte o suficiente para ser encontrado, caso fosse processado todo o conjunto de dados.

Na terceira fase do SEMMA, **modificação**, são criadas, selecionadas, e transformadas as variáveis que serão utilizadas para o processo de construção do modelo. Com base nas descobertas da fase de exploração, pode ser necessário manipular os dados de forma a incluir informação tal como agrupamentos de clientes, sub grupos significativos, ou a introdução de novas variáveis. Pode ser também necessário reduzir o número de variáveis, de forma a restringi-las para apenas as mais significativas.

A quarta fase do processo, é a **modelação**. Nesta fase é procurada uma combinação de variáveis (um modelo), que preveja de forma confiável o resultado pretendido. Depois dos dados terem sido preparados, é necessário construir os modelos que representam padrões nos dados. As técnicas de modelação em *data mining* incluem redes neuronais, modelos baseados em árvores, modelos logísticos e outros métodos estatísticos. Cada técnica de modelação tem os seus pontos fortes, e é apropriada para situações específicas de *data mining*.

A última fase do SEMMA é a **avaliação**. Nesta fase é determinada a utilidade e confiabilidade do modelo obtido a partir do processo de *data mining*. Um método comum de avaliar o modelo é aplicá-lo à partição dos dados para testes criada na fase de amostragem (dados que não foram utilizados na criação do modelo). Se o modelo for válido, deve funcionar para os dados de testes tal como para os dados utilizados para construir o modelo. Similarmente, o modelo pode ser testado com dados conhecidos. Por exemplo, se são conhecidos quais os clientes num ficheiro que têm altas taxas de retenção e o modelo prevê retenção, pode ser verificado se o modelo seleciona os clientes conhecidos.

Ao verificar os resultados de cada fase do SEMMA, é possível determinar como modelar novas questões de negócio que surjam a partir dos resultados obtidos previamente, e assim retroceder para a fase de exploração de forma a realizar um refinamento adicional dos dados.

2.2.3 Comparação entre metodologias

Numa primeira análise a ambas as metodologias, pode-se ter a ideia que a metodologia CRISP-DM é mais completa que o SEMMA. O que não se verifica numa análise mais profunda de ambas as metodologias. As fases de amostragem e exploração da metodologia SEMMA estão contempladas na fase de análise dos dados do CRISP-DM. A fase de modificação do SEMMA é equivalente à fase de preparação dos dados do CRISP-DM.

A principal diferença entre o CRISP-DM e o SEMMA é que o segundo está associado a uma ferramenta específica, o SAS Enterprise Miner, enquanto o CRISP-DM pode ser aplicado com qualquer ferramenta. Outros aspetos diferenciadores entre as metodologias são que o CRISP-DM contém uma fase de análise do problema, enquanto o SEMMA necessita de partir de uma questão de negócio bem definida, não contendo uma fase específica para análise do problema. O CRISP-DM contém também uma fase de integração dos modelos criados com os processos do negócio, não estando contemplada esta fase no SEMMA.

Tabela 1 : Comparação entre metodologias de *data mining*

CRISP-DM	SEMMA
Análise do problema	———
Análise dos dados	Amostragem Exploração
Preparação dos dados	Modificação
Modelação	Modelação
Avaliação	Avaliação
Integração	———

2.3 Metodologias ágeis

Com o desenvolvimento da indústria de software, algumas técnicas para gerir e prever o custo dos projetos foram aparecendo. A metodologia que dominou o desenvolvimento de projetos durante várias décadas foi a metodologia *Waterfall* (Leau et al., 2012). Esta metodologia foi criada em 1970 para descrever um método para a gestão de projetos através dos cinco passos descritos na Figura 3: levantamento de requisitos, planeamento do projeto, implementação, verificação e manutenção.

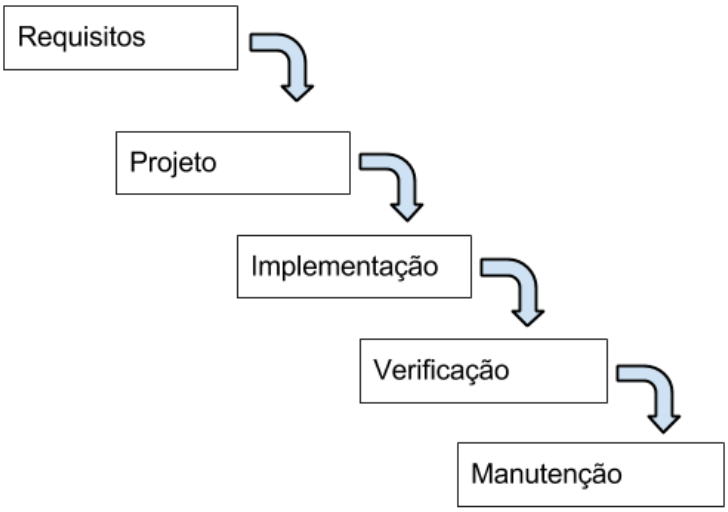

Figura 3 : Ilustração da metodologia *Waterfall*

As metodologias tradicionais de desenvolvimento de software são rígidas e inflexíveis, sendo bastante difícil executar alterações em qualquer fase do projeto. Por essa razão é necessário recolher todos os requisitos numa fase inicial do projeto de forma a evitar ao máximo a necessidade de alterações durante o processo. Estas metodologias têm também o problema da falta de integração dos *stakeholders* no desenvolvimento, e a verificação e teste do produto ser executada numa fase muito tardia do processo. As metodologias ágeis surgem assim, como resposta a estes problemas. Na Tabela 2 são descritos os problemas fundamentais das metodologias tradicionais, e de que forma são solucionados pelas metodologias ágeis.

Tabela 2 : Comparação entre metodologias tradicionais e metodologias ágeis

Metodologias tradicionais	Metodologias ágeis
Quando uma fase é completada, é muito difícil voltar atrás caso seja necessário executar alterações.	Permite que alterações sejam feitas depois do planeamento inicial. As mudanças nos requisitos dos clientes são esperadas.
Alto grau de dependência da qualidade do requisitos iniciais. Caso os requisitos iniciais tenham algum defeito, o projeto pode estar condenado a fracassar.	As alterações nos requisitos são esperadas, é por isso mais fácil adicionar novos requisitos ao produto, melhorando o alinhamento do produto com os objetivos do cliente.
Se um erro nos requisitos é encontrado, ou uma alteração é necessária, o projeto pode ter de ser reiniciado, ou mesmo cancelado.	No final de cada iteração, as prioridades do projeto são avaliadas. Isto permite ao cliente adicionar o seu *feedback* de forma a

	obter o produto que realmente deseja.
Todo o projeto é apenas testado no fim do processo. Caso existam problemas no código, escritos nas fases iniciais, mas descobertos numa fase tardia, a sua existência pode comprometer todo o desenvolvimento realizado a partir desse momento.	São realizados testes no final de cada iteração, garantindo que os erros são capturados e cuidados dentro do ciclo de desenvolvimento. Garantindo assim que não são encontrados apenas no final.
O plano de trabalho não tem em conta alterações nas necessidades do cliente. Se o cliente realizar a meio do processo que necessita mais do que o pensado inicialmente, e exige alterações, muito provavelmente o projeto será entregue mais tarde que o planeado.	Como o produto é testado continuamente e o *feedback* do cliente adicionado ao produto, está em condições de ser lançado no final de qualquer iteração. Aumentando a probabilidade de respeitar a data de lançamento estabelecida.

Uma das diferenças fundamentais entre as abordagens ágeis e as tradicionais é que enquanto as tradicionais incorporam fases distintas, com pontos de controlo e resultados específicos para cada fase, as metodologias ágeis utilizam o conceito de iterações. O resultado de cada iteração pode ser utilizado para avaliar e responder às alterações das necessidades dos clientes. Nas abordagens tradicionais, raramente é entregue um produto final que satisfaça os requisitos do cliente, devido a ser assumido que o cliente tem a perfeita noção dos requisitos que pretende ver implementados desde o momento inicial, o que muitas das vezes não é verdade (Ashmawi, 2013).

Nas metodologias ágeis, as equipas trabalham em conjunto com os demais *stakeholders* de forma a definir o problema a ser resolvido. A equipa define os requisitos para cada iteração, desenvolve o código e os *stakeholders* verificam os resultados no fim da iteração. A verificação dos requisitos ocorre, por isso, muito mais cedo no processo de desenvolvimento do que nos processos tradicionais, onde só acontece no fim do desenvolvimento. Permite assim a alteração de requisitos enquanto ainda são fáceis de alterar (Keith, 2002). Na Figura 4 é ilustrado o conceito de iteração, presente nas metodologias ágeis.

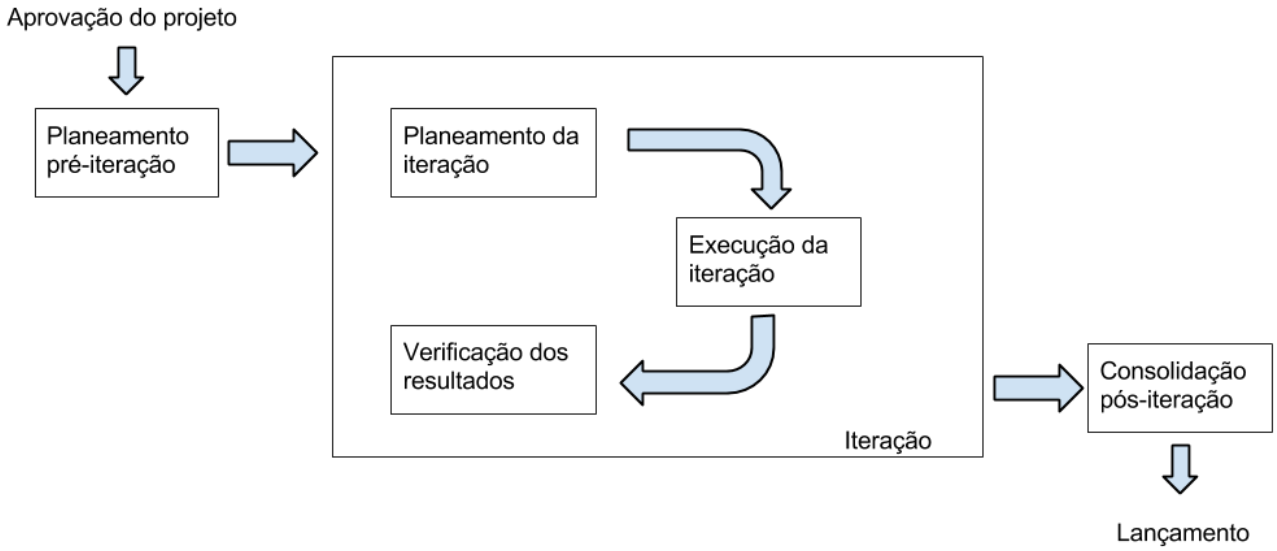

Figura 4 : Ilustração do conceito de iteração numa metodologia ágil genérica

O termo desenvolvimento ágil de software é utilizado por diversas metodologias de desenvolvimento de software. Cada uma é única na sua abordagem específica ao desenvolvimento dos projetos, embora partilhem entre si os 12 princípios estabelecidos no manifesto ágil (Agile Manifesto, 2001). O manifesto ágil é um conjunto de princípios para orientar equipas de desenvolvimento de software. Os 12 princípios ágeis são:

- Prioridade em satisfazer o cliente através da entrega antecipada e contínua de *software*;
- As mudanças nos requisitos são bem vindas, mesmo no final do desenvolvimento;
- Entregar *software* funcional com frequência, a partir de um par de semanas para um par de meses, dando sempre preferência ao período mais curto possível;
- Os clientes e equipas de desenvolvimento devem trabalhar em conjunto diariamente durante o processo;
- Construção de projetos em torno de indivíduos motivados. Dar o ambiente e suporte que necessitam, e confiar que irão obter o trabalho esperado;
- O método mais eficiente e eficaz de transmitir informação para a equipa, e entre a equipa está na conversa cara-a-cara;
- O *software* funcional é a principal medida de progresso;

- Os processos ágeis promovem o desenvolvimento sustentável. Todos os envolvidos no processo devem ser capazes de manter um ritmo constante indefinidamente;
- Atenção contínua à excelência técnica e ao bom planeamento;
- Simplicidade é essencial;
- Os melhores requisitos, arquiteturas e projetos surgem de equipas auto organizadas;
- Em intervalos regulares, a equipa deve refletir sobre como se tornar mais eficaz, ajustando o seu comportamento de acordo com as conclusões geradas.

A filosofia de desenvolvimento ágil tem as suas origens na realidade do mercado atual. É uma tentativa de lidar com os problemas introduzidos pela rápida alteração e imprevisibilidade dos mercados.

Os processos ágeis introduzem a ideia de simplicidade. Nunca produzir mais do que aquilo que é necessário e nunca produzir documentos que tentam prever o futuro. Quanto maior for a quantidade da informação, maior o esforço necessário para encontrar a informação necessária, e maior o esforço para manter a informação atual (Wendorff, 2002).

Os valores derivados da filosofia dos processos ágeis servem para gerir a mudança e reduzir custos. Os elementos das equipas trabalham de forma conjunta, produzindo código em vez de documentação de alta manutenção de forma a aumentar a produtividade. Finalmente, em vez de impor um plano rígido, utilizando metodologias de processos preditivos com exigências rígidas e imutáveis, aceitam a mudança. Reconhecê-la é parte do ambiente atual de negócios e deve-se trabalhar de forma a lidar com as mudanças que ocorrem. Isto não significa que haverá um grupo de programadores a criar o caos sem ter em conta o processo. Os processos ágeis são rigorosos, exigindo trabalho de forma a serem bem sucedidos.

Nos subcapítulos seguintes é feita uma descrição das metodologias ágeis com maior aceitação, o Scrum, Extreme Programming(XP), Adaptive Software Development(ASD), Dynamic Systems Development Methodology(DSDM) e a família de metodologias Crystal. É também realizada uma comparação entre as metodologias tendo em conta diversas dimensões.

2.3.1 Scrum

O Scrum para o desenvolvimento de software foi criado pela comunidade de prototipagem rápida, devido à necessidade de terem uma metodologia que suportasse um ambiente em que os requisitos não eram apenas incompletos no início do projeto, mas onde também poderiam ser alterados rapidamente durante o desenvolvimento (Keith, 2002). A metodologia Scrum inclui processos tanto de gestão como de desenvolvimento.

No centro do scrum está o *backlog* do trabalho a realizar. O *backlog* consiste numa lista de itens priorizados a serem desenvolvidos. O *backlog* é povoado durante a fase de planeamento de um projeto e define os requisitos iniciais, embora possa ser atualizado durante o desenvolvimento.

Depois de uma equipa completar os requisitos do projeto e o desenho de alto nível, o processo de desenvolvimento é dividido numa série de iterações chamados de Sprint*s*. Cada Sprint tem como objetivo a conclusão de um número fixo de pontos do *backlog*. No planeamento do Sprint, a equipa identifica os pontos do *backlog* a serem concluídos. No final do Sprint, a equipa revê o Sprint de forma a verificar o progresso.

Durante um Sprint, a equipa tem um encontro diário, chamadas reuniões de *scrum*. Cada membro da equipa descreve o trabalho que será realizado no dia, o progresso realizado no dia anterior, e os entraves que tenham de ser resolvidos. De forma a manter estas reuniões curtas, todas os envolvidos devem estar na mesma sala, mantendo-se de pé durante toda a reunião.

Quando uma parte significativa do *backlog* estiver implementado e o cliente acreditar que o produto vale a pena colocar em produção, a gestão fecha o desenvolvimento. A equipa começa então a realizar testes de integração e a documentação necessária para o lançamento.

O processo de desenvolvimento do *scrum* consiste em gerir os Sprints. Antes de cada Sprint começar, a equipa planeia o Sprint, identificando os pontos do *backlog* a desenvolver e atribuindo-os a elementos da equipa. Durante o Sprint a equipa desenvolve, revê e ajusta cada um dos pontos do backlog selecionados para o Sprint (Figura 5). Durante o desenvolvimento, a equipa determina as tarefas necessárias para implementar cada ponto do *backlog*. A equipa desenvolve de seguida o código, testa-o, e documenta as alterações. Durante o envolvimento, a equipa cria um executável que é necessário para demonstrar as alterações. Na revisão, a equipa demonstra as novas funcionalidades, adiciona novos pontos ao *backlog,* caso seja necessário, e verifica o risco. Finalmente, a equipa consolida os dados da revisão e ajusta o *backlog* com as alterações necessárias.

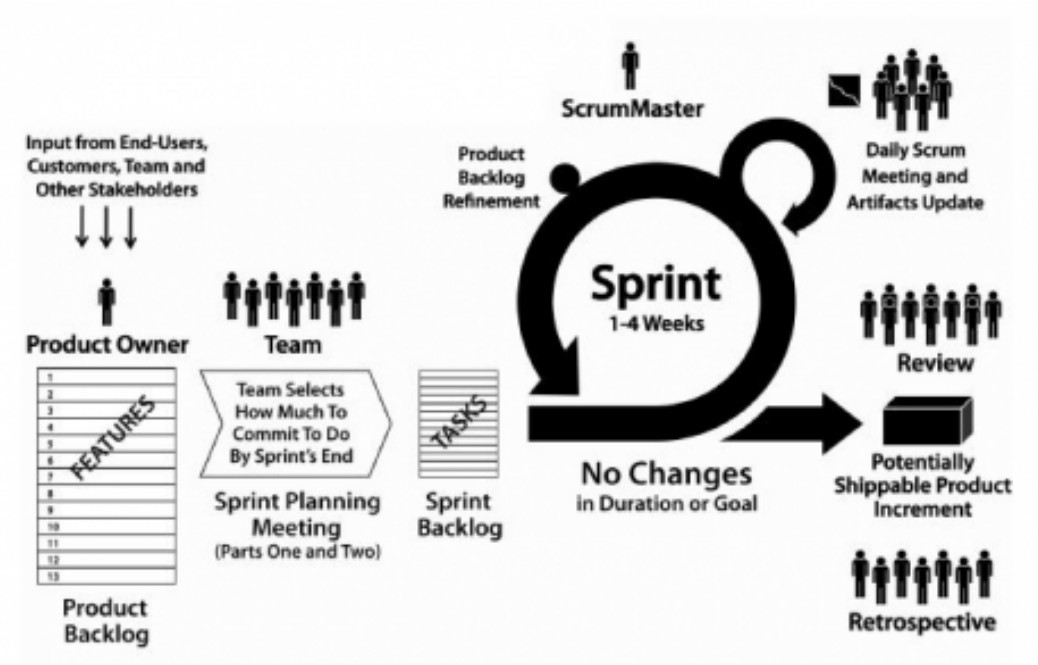

Figura 5 : Metodologia SCRUM

2.3.2 Extreme programming

Extreme programming(XP) é uma metodologia para pequenas e médias equipas de desenvolvimento de software que enfrentam requisitos vagos e em constante alteração. Esta metodologia concentra-se principalmente no desenvolvimento do produto em vez dos aspetos da gestão do projeto, tendo sido criada com o objetivo de ser possível adaptar toda a metodologia ou apenas uma parte (Beck, 1999).

Um projeto em XP começa com uma fase exploratória, que é seguida por uma fase de planeamento, esta fase por sua vez é seguida de um ciclo de iterações, cada uma concluindo com testes de aceitação por parte do cliente. Quando o produto satisfaz o cliente, a equipa termina as iterações e fecha o desenvolvimento (Figura 6).

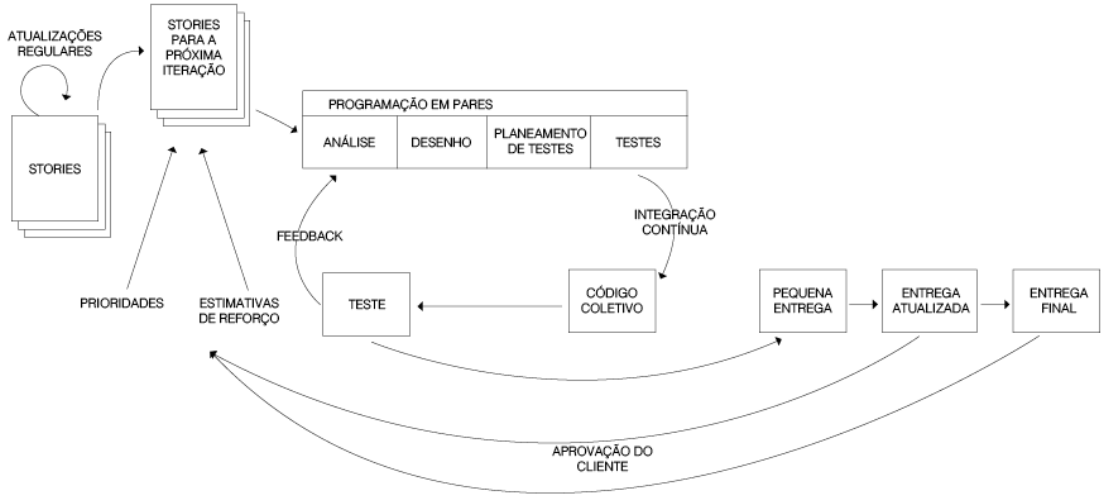

Figura 6 : Fluxo de trabalho da metodologia *Extreme Programming*

Na fase de exploração são criadas *user stories* de forma a descrever os requisitos que o software tem de preencher. As *user stories* ajudam a equipa a determinar o tempo e os recursos necessários para o projeto e a definir os testes de aceitação. O cliente é parte da equipa de XP, de forma a poder adicionar detalhes aos requisitos à medida que o software é criado, permitindo assim aos requisitos evoluir enquanto a equipa de desenvolvimento e o cliente definem como o produto final será.

De forma a planear o projeto, a equipa divide as tarefas de desenvolvimento em iterações. Na fase de planeamento são definidas as tarefas para cada iteração. No fim de cada iteração, o utilizador realiza testes de aceitação contra as *user stories*. Se forem encontrados problemas, resolver esses mesmos problemas passa a ser um passo na próxima iteração (Keith, 2002). Caso o cliente decida que foram completadas suficientes *user stories*, a equipa pode escolher terminar o projeto antes do que foi originalmente planeado.

Alguns conceitos e regras importantes do XP que são necessários ter em conta são:

- Integração contínua, isto é, a equipa de desenvolvimento deve integrar as alterações na base do desenvolvimento pelo menos uma vez por dia.

- Velocidade do projeto. Consiste na medida de que quantidade do trabalho está a ser realizado, sendo importante para o planeamento e escalonamento de atualizações.

- Programação em pares. Este conceito refere que toda a produção do projeto deve ser criada por duas pessoas trabalhando em conjunto em apenas um computador. O XP propõe que duas pessoas a trabalhar em conjunto irão satisfazer as *user stories* que lhes forem atribuídas ao mesmo ritmo que duas a trabalhar em separado, mas com muito mais qualidade.

- Uma *user story*, descreve objetivos a serem resolvidos pelo sistema a implementar. Devem ser escritas pelo cliente e devem ter conter apenas três frases. A *user story* não descreve a solução, nem utiliza linguagem técnica, é uma descrição simples de um requisito para o sistema. Um exemplo para uma *user story* é o seguinte: Como utilizador, quero pesquisar por um produto, de forma a encontrar o artigo que pretendo. Uma *user story* pode ser de dois tipos, um épico ou uma *user story* normal. Uma *user story* é considerada um épico caso exija um esforço de desenvolvimento superior à duração de uma iteração, devendo por essa razão ser decomposta em *user stories* que necessitem de um esforço inferior. Um conjunto de *user stories* relacionadas é chamado de tema.

2.3.3 Adaptive software development (ASD)

A metodologia Adaptive Software Development (ASD), foi desenvolvida por Jim Highsmith e não inclui muitos dos elementos frequentemente associados a uma metodologia. Não é apresentada como uma metodologia para desenvolver projetos de software mas sim como uma abordagem ou atitude que deve ser adotada por uma organização para desenvolvimento ágil de *software*. Elementos genéricos do planeamento de um projeto como marcos, métodos, e entregáveis não são discutidos pela metodologia. O núcleo do ASD é a premissa que não é possível planear com sucesso num ambiente de negócios em constante movimento e imprevisível. O ASD assume a mudança como a norma, estando preparado para a aceitar.

No ASD, o planeamento estático é posto de parte, sendo o ciclo de vida característico de um projeto, isto é, planear, desenhar e construir, substituído por um ciclo de vida dinâmico e adaptativo de especular, colaborar e aprender (Figura 7). É um ciclo de vida dedicado à aprendizagem contínua e orientado para a mudança, reavaliação e intensa colaboração entre elementos da equipa de desenvolvimento, gestores e clientes (Everette R. Keith 2002).

Figura 7 : Ciclo de vida evolutivo vs Ciclo de vida adaptativo

No ciclo de vida adaptativo, a especulação dá espaço para explorar, tornando claro que não existem certezas, não existindo receios de possíveis desvios ao plano. O que não significa que

planear esteja obsoleto; apenas que é um processo ténue. Uma equipa que especula não abandona o planeamento, percebe apenas que existe um grande factor de incerteza nesse processo. A especulação reconhece a natureza incerta de problemas complexos, encorajando por isso a exploração e experimentação.

O segundo componente conceptual do ASD, é a colaboração. Embora exista sempre espaço para melhoria, a maior parte dos elementos das equipas de desenvolvimento de software são razoavelmente proficientes nas atividades de análise, programação e teste. Aplicações complexas requerem que um grande volume de informação seja recolhido, analisado, e aplicado ao problema, um volume que pode ser muito maior do que qualquer individuo possa manipular individualmente. Por exemplo, a construção de um site de comércio electrónico requer uma maior diversidade tanto tecnológica como de conhecimento de negócio do que um projeto do mesmo tipo de à 5 anos atrás. Neste ambiente de grande fluxo de informação, em que um indivíduo ou uma pequena equipa não consegue saber tudo, as técnicas de colaboração, isto é a capacidade de trabalhar em conjunto com o objetivo de produzir resultados, partilhar conhecimento e tomar decisões, são essenciais.

A partir do momento que admitimos que o ser humano é propenso à falha, as práticas de aprendizagem (fase de aprendizagem do ciclo de vida adaptativo), torna-se vital. É necessário por isso testar o nosso conhecimento constantemente, utilizando práticas como por exemplo retrospetivas do projeto e discussões de grupo com os clientes. Mais ainda, devem ser realizadas avaliações no final de cada iteração em vez de esperar pelo final do projeto.

O ciclo de vida adaptativo do ASD tem seis características básicas:

- Foco no objetivo
- Baseado nos resultados
- Iterativo
- Impulsionado pelo risco
- Tolerante à mudança

O ciclo de vida ASD foca-se nos resultados, e não nas tarefas. Os resultados são identificados pelas funcionalidades desenvolvidas numa determinada iteração. Enquanto que os documentos podem ser definidos como resultados, são sempre considerados secundários em relação às funcionalidades de *software* desenvolvidas, uma vez que representam resultados diretos para o cliente. É esperado que o desenvolvimento do produto necessite de várias iterações, enquanto o cliente dá o seu *feedback* das funcionalidades desenvolvidas.

2.3.4 *Dynamic systems development methodology* (DSDM)

A metodologia DSDM surge como uma extensão do RAD (*Rapid Application Development*), focada em projetos de Sistemas de Informação caracterizados por prazos e orçamentos apertados (Ashmawi, 2013). O DSDM consiste em três fases sequenciais: Pré-projeto, Projeto e Pós-projeto (Figura 8).

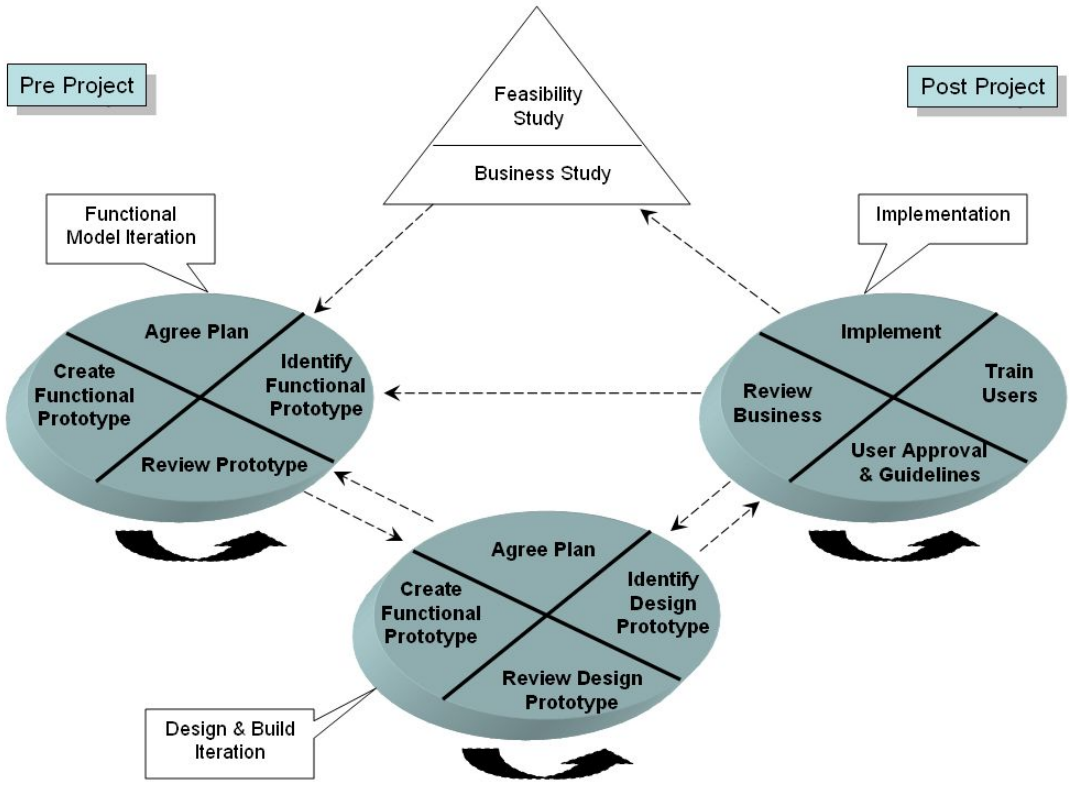

Figura 8 : Ilustração do processo DSDM (M. Cohn, 2004. "*Cycle of DSDM". Selecting an Agile Process: Choosing Among the Leading Alternatives. Proceeding of SD Best pratices*; conference & expo 2004)

Na fase do pré-projeto, o projeto é identificado, definido o plano de financiamento e é assegurado um compromisso de realização. Tratar destas questões numa fase inicial evita problemas em fases mais avançadas do desenvolvimento do projeto.

O ciclo de vida do projeto é composto por cinco fases que a equipa de desenvolvimento terá de percorrer de forma a criar o produto final. Os dois primeiros níveis, o estudo de viabilidade e o estudo de negócio, são fases sequenciais que se complementam. Depois destas fases estarem concluídas, o sistema é desenvolvido iterativamente e de forma incremental nos níveis de análise funcional, desenho e implementação:

- Estudo de viabilidade. Durante este nível do projeto a viabilidade de utilização da DSDM é examinada.

- Estudo do negócio. Incrementa todo o trabalho realizado no estudo de viabilidade. Depois do projeto ser declarado viável, é examinado o processo de financiamento, os utilizadores envolvidos e os seus requisitos. Uma importante propriedade da lista de requisitos determinada é a possibilidade de se definir prioridades. Estas prioridades são definidas utilizando uma perspectiva *MoSCoW (Must, Should, Could, Wont)*.

- Análise funcional. Neste nível os requisitos identificados são convertidos para um modelo funcional. A prototipagem é uma das técnicas chave dentro deste nível, que ajuda no bom envolvimento do utilizador final com o projeto. O protótipo desenvolvido é revisto pelos diferentes *stakeholders*.

- Desenho. O objetivo fundamental desta iteração é a integração dos componentes funcionais desenvolvidos no nível anterior na arquitetura de um sistema que satisfaça as necessidades do utilizador.

- Implementação. Neste nível o sistema é desenvolvido, testado e a sua documentação entregue aos utilizadores finais que deverão ser treinados para a futura utilização do sistema. Nesta fase o sistema a ser entregue foi revisto de forma a responder a todos os requisitos definidos.

A fase de pós-projeto assegura um sistema de atuação eficiente, isto é, um sistema onde está assegurada a sua manutenção e melhoramento de acordo com os princípios da DSDM.

2.3.5 Processos Crystal

A metodologia Crystal é uma das mais leves e adaptáveis abordagens ao desenvolvimento de software. Crystal é na verdade composta por uma família de metodologias ágeis, como o Crystal Clear, Crystal Yellow, Crystal Orange, cujas características únicas, são movidas por determinados fatores, tais como tamanho da equipa, importância do sistema a desenvolver e a prioridade do projeto. Projetos de maior dimensão, que necessitam de maior coordenação e comunicação, são ligados a cores mais escuras (transparente, amarelo, laranja, vermelho, e assim sucessivamente) (Figura 9). A família de metodologias Crystal aborda a ideia de que cada projeto pode necessitar de um conjunto de práticas e processos diferentes de forma a atender às características únicas do projeto.

Os princípios fundamentais da família Crystal incluem o trabalho em equipa, comunicação e simplicidade, bem como a reflexão de forma a ajustar e melhorar o processo. Tal como as outras metodologias ágeis, o Crystal promove a entrega antecipada e frequente de software

funcional, alto envolvimento do cliente, adaptabilidade e remoção da burocracia e das distrações (Strode. 2005).

Selecionar uma metodologia da família Crystal requer que o projeto seja ligado a um de quatro níveis críticos: conforto, dinheiro disponível, dinheiro essencial e vida (Figura 9).

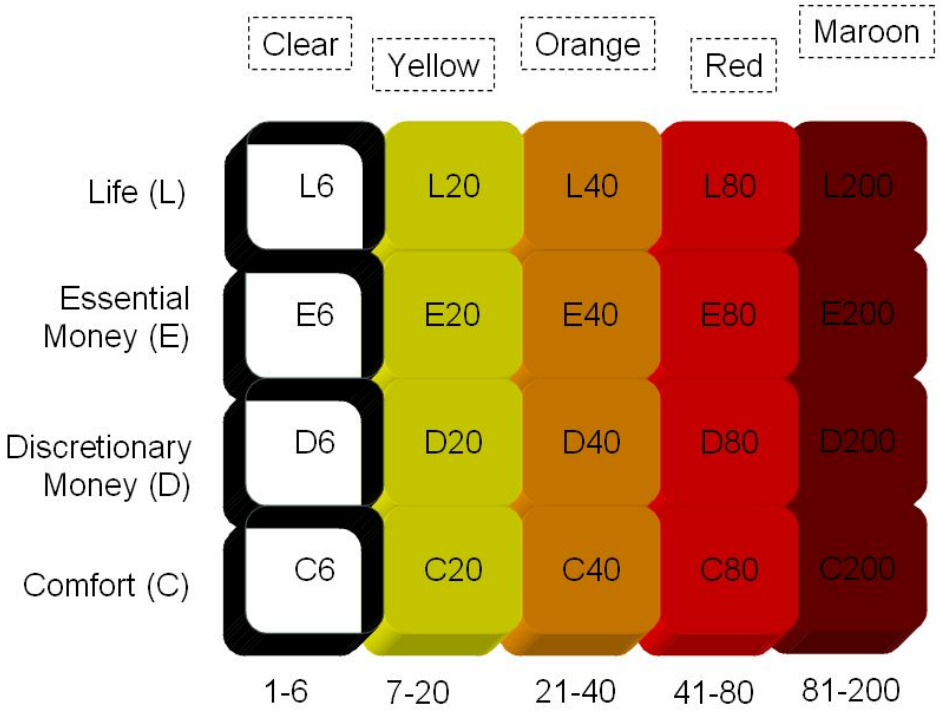

Figura 9 : Variáveis para a seleção de uma metodologia da família Crystal (*A. Cockburn, 2004. "Crystal's coverage of diferente project types". Crystal Clear: A Human-Powered Methodology For Small Teams, including The Seven Properties of Effective Software Projects*, 240)

As metodologias Crystal direcionam as equipas para sete propriedades de segurança, sendo que as primeiras três são fundamentais para o Crystal. As outras quatro podem ser adicionadas de forma a aumentar a margem de segurança no projeto. As propriedades são: entregas frequentes, estreita comunicação entre elementos da equipa, melhorias pela reflexão, segurança pessoal, concentração, fácil acesso a utilizadores experientes, ambiente técnico com testes automatizados e integração frequente.

De forma a seguir uma metodologia Crystal as equipas devem respeitar um conjunto de cinco estratégias. A estratégia de exploração a 360º, isto é, no início de um projeto, geralmente durante a atividade de planeamento, a equipa necessita de estabelecer que o projeto tem valor e que é executável utilizando as ferramentas propostas. É necessário por isso abordar o projeto em todas as direções: valor para negócio, requisitos, domínio, plano ao nível das tecnologias, plano do projeto, equipa de trabalho e o processo a utilizar. O processo exploratório para uma metodologia Crystal pode durar de alguns dias, a semanas.

A segunda estratégia a seguir é a de vitória antecipada. A vitória é uma força que junta equipas e contribui para a auto confiança dos seus membros. Em projetos de software, a vitória antecipada que deve ser procurada pelos membros da equipa, é uma pequena parcela de código funcional e verificado. Isto usualmente consiste numa pequena peça do sistema, muitas vezes não mais do que a possibilidade de adicionar um item à base de dados e depois realizar uma pesquisa pelo mesmo. E muito embora possa não parecer significativa, a partir desta estratégia, os membros da equipa aprendem com estas vitórias o estilo de trabalho de cada um, e os clientes têm uma visão antecipada do sistema.

A terceira estratégia a seguir é a de criação de um protótipo funcional. Este protótipo consiste numa pequena implementação do sistema, geralmente uma função que percorre todo o sistema. Pode não utilizar a arquitetura final, mas deve juntar os principais componentes da arquitetura. A arquitetura e as funcionalidades podem depois evoluir em paralelo.

A quarta estratégia é a de desenvolvimento incremental da arquitetura. A arquitetura do sistema terá de evoluir, a partir do protótipo funcional, e tem também de ser capaz de lidar com alterações nos requisitos e na tecnologia ao longo do tempo. Devido a ser raramente eficaz interromper o desenvolvimento para realizar uma revisão da arquitetura, é aconselhável que a equipa evolua a arquitetura em fases, mantendo o sistema a funcionar ao longo do tempo. A equipa deve aplicar a ideia de desenvolvimento incremental, de forma a refazer a arquitetura, assim como as funcionalidades do sistema.

A última estratégia que deve ser seguida é a utilização de um painel de informação. Isto é, um painel colocado num local onde os elementos da equipa possam ler enquanto trabalham ou se movimentam. Deve mostrar ao leitor informação útil para desenvolvimento, que pode ser necessária, evitando assim a necessidade de questionar. A utilização do painel de informação, significa mais comunicação e menos interrupções.

2.3.6 Comparação entre as diferentes metodologias ágeis

Na Tabela 3 é ilustrada a comparação entre as diferentes metodologias ágeis estudadas. Esta comparação teve em conta sete dimensões, o tamanho da equipa, o período recomendado de uma iteração, a comunicação entre a equipa, a integração com o cliente, a documentação e especificidades de cada metodologia. São também descritas vantagens e desvantagens de cada metodologias (Moniruzzaman e Hossain, 2013). Desta tabela é possível verificar que a metodologia Scrum é a única que não contém especificidades diretamente relacionadas com o desenvolvimento de código para projetos de engenharia de software, sendo uma metodologia mais ligada à gestão do trabalho e da equipa que facilmente pode ser adaptada a outra área que não apenas a engenharia de software (Bass, 2010).

Tabela 3 : Comparação entre metodologias ágeis

Critério	XP	Scrum	ASD	DSDM	Crystal
Tamanho da equipa	Equipa pequena; 2 a 10 elementos	Equipas de pequenas dimensões; 2 a 10 elementos	Suporta qualquer tamanho de equipa	Equipa pequena; 2 a 10 elementos	Suporta qualquer tamanho de equipa
Período recomendado de uma iteração	Uma a seis semanas	Uma a quatro semanas	Quatro a oito semanas	A primeira iteração deve ter um período que permita desenvolver os requisitos críticos do sistema. Para as restantes iterações não é especificada uma duração recomendada.	Até quatro meses para projetos de grandes dimensões
Comunicação entre a equipa	Informal; Reuniões diárias	Informal; Reuniões diárias	Informal;	Formal; Baseada em documentação	Informal;
Integração com o cliente	Cliente é envolvido	Cliente é representado pelo *Product Owner*	Envolvido através dos lançamentos	Envolvido através dos lançamentos	Envolvido através dos lançamentos
Documentação	Apenas documentação básica	Apenas documentação básica	Apenas documentação básica	Documentação complexa	Apenas documentação básica
Especificidades	*User stories, Refactoring*	Sprint, *Product backlog, Spring*	Ciclo de aprendizagem	Prototipagem	Família de métodos adaptativos.

		backlog, Planning Poker, Scrum master			
Vantagens	Espaço de trabalho aberto; Cliente como parte da equipa; Boas práticas; Feedback constante	Elevado nível de comunicação e colaboração entre os elementos da equipa e entre a equipa e o cliente	Desenvolvimento dos componentes de maior risco primeiro; Importância do ciclo de aprendizagem	Gestão eficiente do projeto; Priorização de requisitos	Adaptação ao tipo e tamanho do projeto
Desvantagens	Documentação fraca; Falta de disciplina; Necessária presença do cliente	Documentação fraca; Falta de controlo sobre o projeto;	Falta de documentação;	Documentação complexa;	Altamente teórico; Não define diretrizes para o negócio;

2.4 Conclusões

Com a revisão bibliográfica na área do *data mining* e suas metodologias específicas, e na área das metodologias ágeis da engenharia da software, é possível retirar algumas conclusões quanto à possibilidade de associar as duas áreas.

Das metodologias atuais para o desenvolvimento de projetos de *data mining*, CRISP-DM e SEMMA, a que se demonstra mais indicada para a utilização como base para o desenvolvimento de uma metodologia ágil para projetos de *data mining* é o CRISP-DM, nomeadamente por ser um guia para o processo de *data mining* que não está associado a uma ferramenta específica e por considerar fases essenciais de um projeto de *data mining*, como por exemplo, a compreensão do problema e do negócio, ao contrário do SEMMA.

Quanto às metodologias ágeis, o Scrum, é uma metodologia mais centrada na gestão do trabalho e da equipa, não tendo associadas diretrizes quanto à forma como se deve realizar o desenvolvimento dentro da iteração (Sprint) é a mais aplicável a outras áreas que não apenas a engenharia de software. Alguns conceitos de outras metodologias como as *user stories* e os

testes de aceitação da metodologia Extreme Programming também se demonstram interessantes para aplicar em projetos de outras áreas que não apenas a engenharia de software.

Capítulo 3

Scrum-DM

Neste capítulo é apresentada a metodologia de investigação e a metodologia ágil proposta, ilustrada com exemplos de um caso retrospetivo.

3.1 Metodologia de investigação

De forma a desenvolver uma solução para o problema descrito na Secção 1.1, foi realizada uma adaptação dos conceitos das metodologias ágeis, às metodologias de desenvolvimento para projetos de *data mining*.

A metodologia proposta neste estudo foi desenvolvida em três fases. Primeiro uma versão inicial da metodologia foi desenvolvida com base na revisão da literatura descrita no Capítulo 2 e no acompanhamento de dois projetos de *data mining*. Nesta fase foi também realizado um acompanhamento de um projeto de *Business Intelligence* onde alguns conceitos das metodologias ágeis foram aplicados na gestão do projeto. Este acompanhamento, realizado através da participação nas reuniões dos projetos utiliza a observação como método de recolha de dados, de forma a entender como é desenvolvido um projeto de *data mining*. Este estudo preliminar teve por base dois casos: participação do *Data Intelligence Group* da FEUP numa competição de *data mining*, e um projeto de *data mining* do INESC TEC. O Anexo A apresenta um resumo da evidência recolhida nesta primeira fase.

A versão inicial da metodologia foi validada e melhorada recorrendo ao método de caso retrospetivo (Yin, 2003). Foram realizados dois estudos de caso (Tabela 4) e o método de recolha de dados utilizado foi a entrevista. Para estes três casos foram conduzidas 6 entrevistas, tendo sido realizadas durante o mês de Maio. As entrevistas começavam com uma breve introdução ao objetivo do trabalho de investigação. As entrevistas foram gravadas, permitindo

ao investigador focar-se completamente na entrevista. A gravação do áudio foi realizada com o acordo dos entrevistados. A entrevista era composta por duas partes. A primeira parte era relacionada com o projeto de *data mining* em que o entrevistado participou. A segunda, com a metodologia utilizada no desenvolvimento desse projeto. Na primeira parte da entrevista o objetivo era entender quais os objetivos do projeto, como é que esses objetivos foram definidos e se existiu colaboração entre a equipa de desenvolvimento e o negócio para a definição dos mesmos. A segunda parte tinha como objetivo recolher dados acerca da metodologia aplicada no desenvolvimento do projeto. O Anexo B apresenta o guião utilizado para conduzir as entrevistas.

Tabela 4 : Casos de estudo da aplicação retrospetiva da metodologia

Caso	Industria	Objetivos	Número de elementos da equipa	Número de entrevistas
A	Têxtil	Desenvolvimento de um sistema de medição de desempenho da cadeia de logística e um sistema de recomendações com base nos dados recolhidos por etiquetas RFID.	4	3
B	Retalho	Sistema de apoio à decisão que sugira, tendo em conta uma loja, como distribuir o espaço pelas categorias de produto.	3	3

Terceiro, a fim de validar a aplicabilidade real da metodologia, o método de investigação por ação foi aplicado a dois projetos (Gummenson, 2000). Aqui o investigador assumiu o papel de *Scrum Master* para assegurar a aplicação correta da metodologia. No final deste projeto foi realizada uma entrevista a cada elemento da equipa de projeto, tendo sido realizadas 6 entrevistas. Estas entrevistas tinham como objetivo determinar o grau de aceitação pela equipa da metodologia proposta e quais as principais dificuldades e impedimentos da adaptação. O Anexo C apresenta o guião utilizado para conduzir estas entrevistas.

3.2 Metodologia Scrum-DM

A metodologia Scrum-DM consiste numa associação da metodologia ágil Scrum como metodologia para a gestão do trabalho, e da metodologia Crisp-DM como estratégia a seguir no desenvolvimento de um projeto de *data mining*.

O esquema da metodologia (Figura 10), tal como o conceito da metodologia, consiste numa associação do esquema da metodologia Scrum com o esquema da metodologia CRISP-DM. O CRISP-DM foi dividido e adaptado ao Scrum, sendo o Scrum-DM, à semelhança do CRISP-DM iniciado por uma fase de *Business Understanding*, onde é realizada a análise dos objetivos e do negócio, e finalizado por uma fase de *Deployment,* onde é realizada a integração dos resultados do *data mining*. O desenvolvimento é realizado na fase de Sprint, estando contempladas as fases de *Data Understanding, Data Preparation, Modeling e Evaluation* do CRISP-DM.

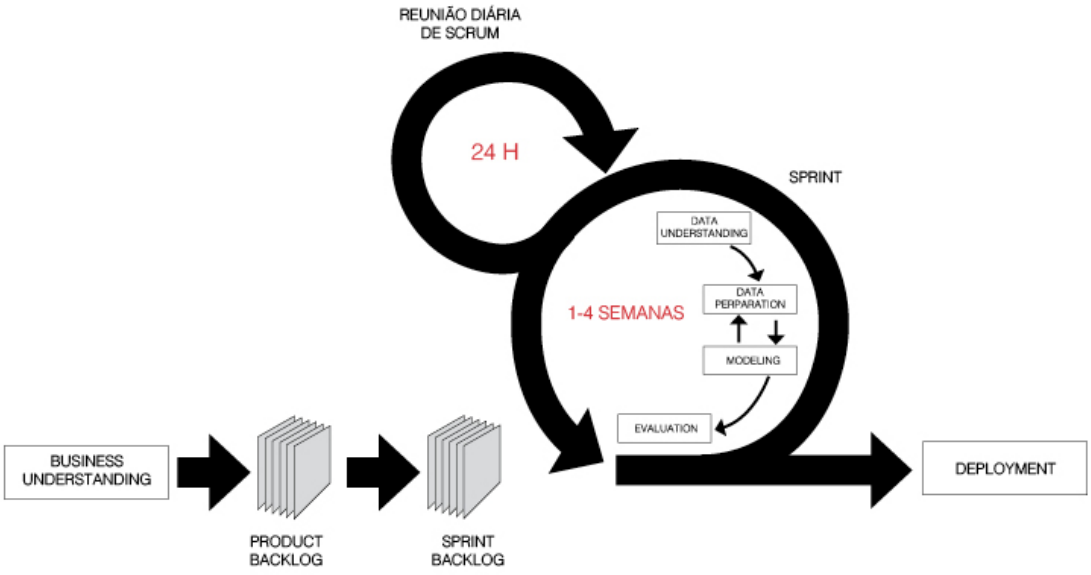

Figura 10 : Metodologia Scrum-DM

Para a aplicação da metodologia é necessário definir quatro conceitos, *Data mining Story* (ver seção 4.1.1.2), *Product Owner, Scrum Maste*r e equipa de desenvolvimento (ver seção 4.1.1.1). É necessário também existirem dois documentos, o *Product Backlog* e o Sprint *Backlog*. O *Product Backlog* consiste numa lista priorizada de *Data mining Stories* e outros

requisitos a serem desenvolvidos no projeto, representando a qualquer momento a visão de tudo o que pode ser realizado. O outro documento necessário é o Sprint *Backlog*. Para cada Sprint existe um Sprint *Backlog*, contendo uma lista de todas as tarefas necessárias para desenvolver os itens do *Product Backlog* escolhidos para o Sprint. Esta lista de tarefas é criada pela equipa, decompondo cada item selecionado do *Product Backlog* em tarefas que serão executadas durante o Sprint. Esta decomposição em tarefas deve ter em conta as tarefas da metodologia Crisp-DM, e os objetivos do item do *Product Backlog*. Tantos os itens do *Product Backlog* como do Sprint *Backlog* devem ser estimados pela equipa.

Durante o Sprint, existem cinco reuniões essenciais para o correto funcionamento da metodologia. Uma reunião de planeamento do Sprint, reuniões diárias de Scrum, uma reunião de refinamento do Sprint, e no final do Sprint, as reuniões de revisão e retrospetiva.

De forma a exemplificar a aplicação prática de alguns conceitos da metodologia desenvolvida, foi feito um estudo de caso da aplicação retrospetiva da metodologia num projeto de *data mining* na área do retalho (Pinto e Soares, 2012). Este caso aborda o problema de atribuir espaço a categorias de produtos em lojas de retalho. Dado que o espaço é um recurso limitado, é essencial para o retalhista prever os efeitos da alocação de mais espaço a uma categoria em vez de atribuir a uma outra categoria de produto. Este projeto tinha como objetivo desenvolver um sistema de apoio à decisão para alocação do espaço que combinasse técnicas de *machine learning* com um método meta-heurístico de otimização.

3.2.1 Descrição de conceitos

Para a metodologia ser aplicável é necessário definir alguns conceitos fundamentais. Os papéis necessários para a metodologia e as *Data Mining Stories*.

3.2.1.1 Papéis

Product Owner

O *Product Owner* tem um papel decisivo no projeto. É responsável por maximizar o retorno sobre o investimento, identificando as *Data Mining* Stories, traduzindo-as para uma lista priorizada, decidindo quais as que devem estar no topo da lista para o próximo Sprint. Durante o projeto deve continuamente priorizar e refinar a lista de *Data Mining* Stories. É também quem valida os critérios de aceitação para cada *Data Mining Story*.

O *Product Owner* é responsável por escolher para cada Sprint os itens com maior valor para o cliente. Em alguns casos, o *Product Owner* e o cliente são a mesma pessoa. Isto é comum para aplicações internas. O papel do *Product Owner* é diferente do tradicional gestor de produto pois interage ativamente e regularmente com a equipa, prioriza trabalhando em conjunto com todos os stakeholders e revê os resultados de cada Sprint, em vez de delegar as decisões de

desenvolvimento para um gestor de projeto. Deve existir apenas uma só pessoa que serve e tem a responsabilidade de *Product Owner*, embora não tenha necessariamente de trabalhar sozinho.

O *Product Owner* para ser bem sucedido deve entender perfeitamente o cliente, e idealmente deve ter conhecimento da abordagem necessária para o desenvolvimento de um projeto de *data mining*.

O *Scrum Master*

O *Scrum Master* tem o papel de guia para a equipa e para o *Product Owner,* assegurando que a metodologia é aplicada corretamente. Deve fazer tudo o que estiver ao seu alcance para ajudar a equipa, o *Product Owner* e a organização a terem sucesso. O *ScrumMaster* não é um gestor do projeto ou o representante da equipa. O seu objetivo é servir a equipa, ajudando a eliminar obstáculos e protegendo-a de interferências externas. Tem também a responsabilidade de guiar o *Product Owner* na utilização da metodologia.

Como a metodologia torna visíveis muitos obstáculos e perigos para a eficácia da equipa, é necessário ter um *Scrum Master* a trabalhar de forma energética para ajudar a resolver esses problemas, caso contrário, a equipa ou o *Product Owner* podem encontrar dificuldades. Deve existir um *Scrum Maste*r a trabalhar a tempo inteiro, embora equipas pequenas possam ter um elemento a desempenhar este papel, sendo que neste caso deve ter uma menor carga de trabalho.

O *Scrum Master* e o *Product Owner* não podem ser a mesma pessoa, devido a terem interesses diferentes. A combinação dos dois papéis na mesma pessoa pode gerar confusões e conflitos. O *Scrum Maste*r não diz à equipa o que fazer, ou atribui tarefas, deve apenas facilitar o processo, apoiando a equipa enquanto se organiza e faz a sua gestão.

A equipa

A equipa, ou equipa de desenvolvimento, desenvolve o produto que o *Product Owner* define. A equipa deve ser multifuncional, isto é, deve possuir o conhecimento necessário para entregar um resultado demonstrável ao cliente no final de cada Sprint, possuindo também capacidade para se gerir a si própria, com um elevado grau de autonomia e responsabilidade.

É a equipa que decide quantas *Data Mining* Stories da lista priorizada pelo *Product Owner* irá desenvolver num Sprint, e qual a melhor forma de atingir esse objetivo. Enquanto a equipa desenvolve o produto, deve ao mesmo tempo, fornecer ideias ao *Product Owner* de forma a tornar o produto melhor. A participação de elementos da equipa em múltiplos produtos ou projetos, é desaconselhada, de forma a evitar dividir as atenções e alternância entre contextos.

Para a metodologia ser aplicável a equipa de desenvolvimento deve ser constituída por pelo menos dois elementos e não exceder os dez, trabalhando em conjunto durante cada Sprint de forma a atingir os objetivos

A Tabela 5 sintetiza as responsabilidades de cada papel.

Tabela 5 : Responsabilidades por papel

Papel	Responsabilidades
Product Owner	Identificar e clarificar as necessidades dos *stakeholders;* Representar os *stakeholders;* Definir objetivos e requisitos para o projeto, e colocá-los na forma de *Data Mining Stories;* Gerir o *Product Backlog* durante todo o projeto; Definir o objetivo para cada Sprint; Estar disponível para a equipa; Respeitar os limites de cada Sprint; Rever os resultados de cada Sprint.
Scrum Master	Guiar a equipa e o *Product Owner* na aplicação da metodologia; Ser o especialista na aplicação da metodologia; Ajudar a remover impedimentos que a equipa encontre no desenvolvimento; Proteger a equipa de influências externas; Apoiar a equipa enquanto se organiza a si mesma; Facilitar a aplicação da metodologia; Encorajar a equipa a melhorar as práticas de desenvolvimento.
Equipa	Apoiar a definição das *Data Mining* Stories; Definir e comprometer-se com as tarefas para cada Sprint; Comprometer-se com a entrega do objetivo do Sprint, respeitando os critérios de aceitação; Desenvolver o que está especificado no *Product Backlog*.

3.2.1.2 *Data Mining Stories*

Uma *Data Mining Story* (DMS), é uma adaptação das *user stories*, popularizadas na engenharia de software, ao *data mining*. Ajudam a determinar o tempo e os recursos necessários para o projeto, e a definir os critérios de aceitação.

As *Data Mining* Stories descrevem de forma sucinta e de alto nível, os objetivos para o sistema a implementar. Devem ser escritas na segunda parte da fase de *Business Understanding* pelo *Product Owner*, em conjunto com o *Scrum Master*. Uma *Data Mining Story* deve constituir uma frase, definida por três partes fundamentais.

Como (perfil), quero (objetivo para o *data mining*), de forma a (objetivo para o negócio).

A primeira parte da *Data Mining Story* representa o perfil de utilizador para o qual a mesma trará valor, a segunda representa o objetivo na perspetiva da abordagem de *data mining* e a última representa o objetivo em termos do negócio.

Tal como nas *user stories* tradicionais da metodologia XP descrita na Secção 2.2.2, uma DMS pode ser de dois tipos, épico ou normal. Um épico é uma DMS de grande dimensão, englobando geralmente grande parte da visão para o sistema a implementar. Um épico é geralmente grande demais para ser implementado sem existir uma divisão. Uma DMS normal é caracterizada pelo facto de ser pequena o suficiente para ser divida em trabalho que encaixe na duração de um Sprint. Uma coleção de DMS relacionadas é chamado de tema. As DMS não descrevem a solução, nem é utilizada linguagem técnica da forma tradicional para a definição de requisitos, é na verdade uma descrição simples de um requisito ou funcionalidade (Tabela 6).

Uma DMS só está completa quando a si está associado um ou mais critérios de aceitação. Estes critérios devem ser definidos pelo *Product Owner* em conjunto com a equipa. O critério de aceitação para uma DMS leva a que todo desenvolvimento seja conduzido por ele, pois só quando o critério de aceitação estiver completado é que a equipa pode dar como finalizada a DMS.

Os critérios de aceitação são definidos para cada DMS pelo *Product Owner* em conjunto com a equipa de desenvolvimento após a definição inicial das DMS. Os critérios de aceitação permitem à equipa ter conhecimento durante o Sprint, que o que estão a desenvolver está de acordo com o que é esperado pelo *Product Owner*.

Uma DMS pode ter um ou mais critérios de aceitação. Devem ser definidos os que forem necessários para assegurar que está completa. Cada critério representa um resultado esperado pelo *Product Owner* para a *Data Mining Story*. O *Product Owner* é responsável pela verificação de que os critérios estão corretos e pela revisão dos resultados, de forma a verificar se foram cumpridos. No caso de existirem critérios falhados, o *Product Owner* deve decidir quais os de maior prioridade. Uma DMS não é considerada acabada enquanto não passar o critérios ou critérios de aceitação (Tabela 6).

Tabela 6 : Exemplo de *Data Mining Stories* com critérios de aceitação

Exemplo de critérios de aceitação para as Data Mining Stories identificadas no caso retrospetivo

Epic	Critério de aceitação
Como gestor operacional, quero um sistema de apoio à decisão que me permita otimizar o *layout* das lojas, de forma a maximizar as vendas.	Vendas globais previstas em dados históricos superiores às observadas.

Data Mining Stories	Critério de aceitação
Como analista, quero selecionar um subconjunto de categorias de forma a maximizar a probabilidade de obter modelos de previsão de vendas com boa qualidade.	Seleção de algumas das melhores e das piores categorias em relação à diversidade de espaço de prateleira e de vendas.
Como analista, quero modelos que relacionem o espaço atribuído a uma categoria às vendas dessa categoria, de forma a alimentar o mecanismo de otimização do *layout*.	Erro dos modelos inferior a 10%.
Como analista, quero estudar o desempenho de meta heurísticas de otimização com base nos modelos de previsão, de forma a maximizar as vendas.	Vendas previstas com dados históricos superiores às vendas registadas.

3.2.2 Fases

Nesta Secção são descritas as três fases da metodologia Scrum-DM. As fases de *Business Understanding,* Sprint e *Deployment*.

3.2.2.1 Business Understanding

A fase de *Business Understanding* de um projeto que adopte a metodologia Scrum-DM, é dividida em três partes. Uma primeira parte semelhante à fase de *Business Understanding* da metodologia CRISP-DM, apresentada na Secção 2.1.1.1, onde o *Product Owner* define os objetivos e requisitos do projeto. Tem também uma segunda parte onde são definidas as *Data Mining* Stories e uma terceira onde se associa a cada DMS um critério de aceitação (Figura 11).

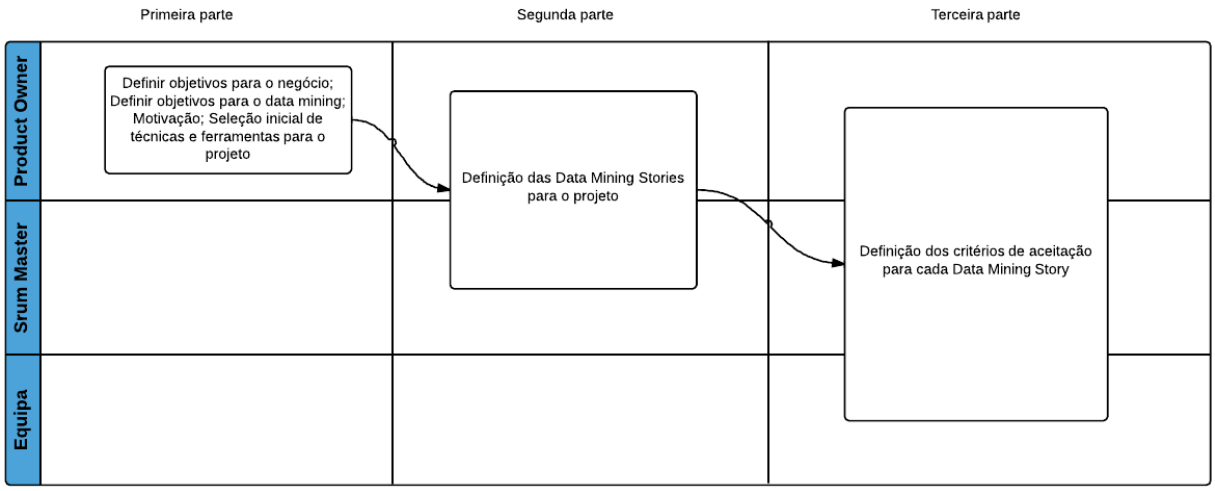

Figura 11 : Fluxo de trabalho na fase de *Business Understanding* da metodologia Scrum-DM

A primeira parte da fase de *Business Understanding* tem como objetivo perceber os objetivos do projeto e requisitos numa perspetiva do negócio, e assim converter este conhecimento na formulação de um problema de *data mining*.

O *Product Owner*, caso não seja ele próprio o cliente, deve entender a fundo, partindo da perspetiva do negócio, o que o cliente realmente deseja realizar. Nesta fase inicial do projeto é essencial descobrir fatores importantes, que possam influenciar o resultado final do projeto.

Na primeira parte da fase de *Business Understanding*, deve existir uma investigação detalhada de todos os recursos, restrições, suposições e outros fatores que devem ser tidos em conta na determinação dos objetivos para as fases posteriores do projeto. É essencial também determinar os objetivos para o *data mining*. Estes objetivos consistem nos objetivos em termos técnicos para o projeto. Um exemplo de um objetivo para o *data mining* é ”prever o resultado nas vendas da alteração do espaço de uma categoria de produtos”, o objetivo para o negócio é “maximizar as vendas”. Nesta fase deve ser também realizada a seleção inicial de técnicas e ferramentas para o projeto.

Quando os objetivos para o *data mining* estiverem claros para o *Product Owner*, é iniciada a segunda parte da fase de *Business Understanding*. Na segunda parte, o *Product Owner*, em conjunto com o *Scrum Master*, define as *Data Mining* Stories para o projeto. A presença do *Scrum Master* é necessária de forma a garantir que a estrutura das DMS é cumprida. As DMS definidas nesta fase devem ser de alto nível, devendo estar representado todo o projeto de uma forma geral, sendo depois decompostas em conjunto com a equipa na definição inicial do *Product Backlog*, e posteriormente de forma iterativa nas reuniões de planeamento de Sprint.

A terceira parte da fase de *Business Understanding*, consiste na definição dos critérios de aceitação para as *Data Mining* Stories definidas. Esta parte deve ser realizada pelo *Product Owner* em conjunto com a equipa de desenvolvimento. Quando todas as DMS definidas pelo

Product Owner estiverem associadas a um ou mais critérios de aceitação, é iniciada a definição do *Product Backlog*.

3.2.2.2 Sprint

O Sprint é a unidade de desenvolvimento iterativo do Scrum. A duração do Sprint é definida antes de ser iniciado o desenvolvimento do projeto, tendo geralmente a duração de uma a quatro semanas. Deve ser preferida sempre a menor unidade de tempo possível para o projeto, que permita no final do Sprint ter um resultado passível de ser mostrado ao cliente.

Cada Sprint é iniciado por uma reunião de planeamento, onde as tarefas para o Sprint são identificadas e estimadas pela equipa, e é finalizado por uma reunião de revisão e uma de retrospetiva, onde o progresso é avaliado e as lições aprendidas com o Sprint identificadas. Durante o Sprint, existe também uma reunião diária, chamada de reunião diária de Scrum, que tem como o objetivo assegurar que a equipa está no caminho correto para atingir os objetivos a que se propuseram para o Sprint. No final de cada Sprint, deve ser possível demonstrar ao *Product Owner* um resultado que esteja finalizado, isto é, testado e documentado caso seja necessário, e que tenha em conta os critérios de aceitação das *Data Mining Stories* do Sprint.

3.2.2.3 Deployment

A fase de *Deployment* da metodologia Scrum-DM é semelhante à do CRISP-DM, com a diferença que só é executada quando o *Product Owner* considerar que estão cumpridos elementos suficientes do *Product Backlog* para satisfazer os objetivos e requisitos do cliente.

A fase de *Deployment* consiste na integração dos resultados do *data mining* com os processos de negócio. Esta fase pode conter quatro tarefas. A primeira tarefa do *Deployment* é o planeamento. Aqui é definida a estratégia de integração dos modelos e resultados do *data mining*.

A segunda tarefa é o planeamento da monitorização e manutenção. A monitorização e manutenção são assuntos de relevo caso os resultados obtidos com o *data mining* passem a ser integrados no dia-a-dia do negócio e do seu ambiente. A preparação cuidada da estratégia de manutenção ajuda a evitar períodos longos e desnecessários de utilização incorreta dos resultados do *data mining*. Por esta razão é necessário definir um plano de monitorização detalhado.

Na fase de *Deployment* pode ser também produzido um relatório final que pode ser apenas um resumo do projeto e suas experiências, no caso de ainda não terem sido documentadas continuamente durante o desenvolvimento, ou pode ser uma apresentação crítica dos resultados do *data mining*.

A última tarefa do *Deployment* consiste na revisão do projeto. Avaliar o que correu bem e o que correu mal, o que foi bem feito e o que necessita de ser melhorado no processo.

3.2.3 Documentos

Nesta secção são descritos os documentos necessários para a aplicação da metodologia. O *Product Backlog* e o Sprint *Backlog*.

3.2.3.1 Product Backlog

Um *Product Backlog*, consiste numa lista priorizada de recursos centrados no cliente. O *Product Backlog* existe e evolui durante a duração do projeto, sendo que representa, em qualquer momento, a visão de tudo o que pode ser realizado pela equipa, por ordem de prioridade. Por cada projeto existe apenas um *Product Backlog*, isto significa que o *Product Owner* é necessário para realizar decisões de priorização durante todo a vida do projeto, representando os interesses dos *stakeholders*, incluindo a equipa.

O *Product Backlog* contém essencialmente novas funcionalidades para o cliente no formato de *Data Mining* Stories, mas também pode conter objetivos de melhoria, como aumentar a velocidade, ou trabalho de investigação, como investigar soluções para melhorar o tempo de processamento de um algoritmo. Pode também conter erros que tenham sido identificados mas ainda não resolvidos.

Os itens do *Product Backlog* devem ser priorizados. Geralmente os itens com maior prioridade devem ser os que entregam maior valor para o cliente naquele momento, com o menor custo. Os itens com maior risco devem ter também uma prioridade alta de forma a impedir que se tornem num bloqueio para o projeto. Um bom *Product Backlog* deve ter os itens de maior prioridade mais detalhados que os de menor prioridade, uma vez que os primeiros serão desenvolvidos mais cedo que os últimos.

De forma a responder à variabilidade e aprendizagem continua durante um projeto, o *Product Backlog* é continuamente atualizado pelo *Product Owner* de forma a refletir as alterações nas necessidades do cliente, novas ideias ou impedimentos técnicos que tenham surgido. É possível em cada Sprint, adicionar, remover, modificar, dividir ou alterar a prioridade dos itens.

Todos os itens do *Product Backlog* necessitam de ser estimados quanto ao esforço necessário para os desenvolver. A cada Sprint, essas estimativas devem ser re-avaliadas com a informação recolhida e aprendida do Sprint anterior. A equipa estima o esforço para cada item e entrega as estimativas ao *Product Owner* (Tabela 7).

Tabela 7 : Exemplo de um *Product Backlog*

Prioridade	*Data mining Story*	Estimativas iniciais
1	Como analista, quero selecionar um subconjunto de categorias de forma a maximizar a probabilidade de obter modelos de previsão de vendas com boa qualidade.	132 horas
2	Como analista, quero modelos que relacionem o espaço atribuído a uma categoria às vendas dessa categoria, de forma a alimentar o mecanismo de otimização do *layout*.	162 horas
3	Como analista, quero estudar o desempenho de meta heurísticas de otimização de vendas com base nos modelos de previsão, de forma a maximizar as vendas.	132 horas

Para cada Sprint deve ser controlada a quantidade de trabalho realizado. Por exemplo, 26 horas de trabalho completadas em média por Sprint. Esta medida pode ser utilizada para projetar uma data de entrega para um certo número de itens do *Product Backlog*, ou estimar a quantidade de itens que podem ser completados até uma determinada data, se a média continuar e nada for alterado. A esta média chama-se velocidade. Esta medida é expressa na mesma unidade que as estimativas para o *Product Backlog*.

Os itens do *Product Backlog* podem variar em tamanho ou esforço. Os de maior dimensão podem ser divididos em unidades mais pequenas durante o refinamento do *Product Backlog* ou na reunião de planeamento do Sprint. Os de menor dimensão podem ser consolidados. Devem ter granularidade suficiente de forma a serem compreendidos pela equipa, permitindo estimar o esforço na reunião de planeamento do Sprint.

Um item do *Product Backlog* pode ser dado como finalizado pela equipa, quando passar o critério de aceitação.

3.2.3.2 Sprint *Backlog*

O Sprint *Backlog* é uma lista de tarefas a serem completadas durante o Sprint. Esta tarefas são identificadas pela equipa de desenvolvimento durante a reunião de planeamento do Sprint. Durante esta reunião, a equipa seleciona um número de DMS do *Product Backlog*, e identifica as tarefas necessárias para completar cada uma. Para cada tarefa a equipa deve estimar o esforço necessário para a completar (Tabela 8). É essencial que seja a equipa a selecionar as tarefas e o

tamanho *do* Sprint *Backlog*, pois essas tarefas representam o compromisso da equipa para o Sprint.

Tabela 8 : Exemplo de um Sprint *Backlog* no início do Sprint

Foi selecionada a primeira DMS do *Product Backlog* para este Sprint, e a duração de quatro semanas.

DMS	Tarefa	Estimativa inicial (em horas)	Novas estimativas de trabalho restante no final de cada dia						
			1	2	3	4	5	6	...
1	Recolha inicial de dados	16							
1	Limpeza dos dados	20							
1	Análise exploratória dos dados	20							
1	Seleção dos dados e das variáveis a utilizar no projeto	20							
1	Definir indicadores da representatividade dos dados das categorias	24							
1	Encontrar categorias com maior volatilidade na área de venda	32							

O Sprint *Backlog* é um documento dinâmico. Durante o Sprint, os membros da equipa devem atualizar o Sprint *Backlog* com nova informação que vá surgindo. Esta é uma atividade que pode ser executada pela equipa durante a reunião diária de *Scrum*. Em cada dia, a equipa deve estimar o esforço restante para cada tarefa do Sprint. A recolha destes dados deve ser feita pelo *Scrum Master*. Com estes dados pode ser criado um gráfico de *BurnDown*. Durante o Sprint, caso a equipa tenha estimado mal a quantidade de trabalho para o Sprint, pode ser necessário adicionar ou remover tarefas.

3.2.4 Reuniões

Nesta secção são descritas as reuniões necessárias numa aplicação da metodologia.

3.2.4.1 Planeamento do Sprint

A reunião de planeamento do Sprint é tipicamente dividida em duas partes. Uma parte onde é definido o que se vai realizar durante o Sprint e outra onde se define como se vai realizar. Na primeira parte devem participar todos os elementos da equipa de *Scrum*. Na segunda devem participar a equipa e o *Scrum Master*, sendo opcional a presença do *Product*

Owner. Cada parte deve ter uma duração máxima de 2h, não devendo exceder as 4h no total das duas partes.

Na primeira parte da reunião de planeamento do Sprint, o *Product Owner* em conjunto com a equipa revê os itens do *Product Backlog* com alta prioridade e que deseja ver implementados no Sprint. A equipa e o *Product Owner* devem discutir os objetivos, o contexto e os critérios de aceitação para cada item. A equipa e o *Product Owner* devem definir o objetivo para o Sprint. Este objetivo deve consistir numa descrição sumária do objetivo do *Product Owner* para o Sprint. Este objetivo dará à equipa alguma flexibilidade em relação ao que podem entregar, pois embora possam remover ou alterar alguma tarefa do Sprint, devem sempre comprometer-se a entregar algo dentro do âmbito do objetivo do Sprint. Nesta primeira parte da reunião a equipa deve estar focada em entender o que o *Product Owner* pretende e porquê. No fim da primeira parte o *Product Owner* pode se retirar, embora deva estar disponível para retirar dúvidas durante a segunda parte.

A segunda parte da reunião de planeamento do Sprint tem como objetivo definir como implementar os itens que a equipa selecionou para o Sprint. A equipa prevê a quantidade de itens que pode completar até ao final do Sprint, começando pelo topo do *Product Backlog* e percorrendo a lista por ordem descendente. É a equipa que decide a quantidade de trabalho que irá completar. A equipa pode retirar itens de menor prioridade do *Product Backlog*, quando o *Product Owner* e a equipa estiverem de acordo que encaixam facilmente com um outro de maior prioridade.

De forma a guiar a tarefa de estimação do trabalho, as equipas podem utilizar a velocidade de Sprints anteriores, ou calcular a sua capacidade. A velocidade do Sprint consiste na quantidade de trabalho que a equipa consegue concluir por Sprint. No caso do cálculo da capacidade, no planeamento do Sprint, a equipa calcula quanto tempo cada membro tem para trabalho relacionado com o Sprint. A maioria das equipas assume que os membros realizam 4 a 6 horas por dia de trabalho relacionado com o Sprint. Depois da capacidade estar determinada, a equipa decide quantos itens do *Product Backlog* podem completar na capacidade definida e de que forma os irão completar, sendo os itens selecionados por ordem decrescente de prioridade até que a capacidade seja preenchida. Os itens escolhidos são decompostos em tarefas a serem realizadas durante o Sprint. Esta lista de tarefas é chamada de Sprint *Backlog*.

A decomposição de uma *Data mining Story* em tarefas deve ter em conta o objetivo para o *data mining* da DMS selecionada e as fases do CRISP-DM necessárias para o seu desenvolvimento (Tabela 9). As fases a ter em conta são, *Data Understanding, Data Preparation, Modeling* e *Evaluation*. As tarefas de cada fase do CRISP-DM necessárias para completar uma *DMS,* são descritas na Secção 2.1.1.1, e devem ser tidas em consideração, como um guia, para a decomposição de uma DMS em tarefas.

Tabela 9 : Exemplo de Sprint *Backlog* para três Sprints com a duração de quatro semanas

Sprint 1

Item do *Product Backlog*	Tarefa	Estimativa inicial (em horas)
1	Recolha inicial de dados	16
1	Limpeza dos dados	20
1	Análise exploratória dos dados	20
1	Seleção dos dados e das variáveis a utilizar no projeto	20
1	Definir indicadores da representatividade dos dados das categorias	24
1	Encontrar categorias com maior volatilidade na área de venda	32

Sprint 2

Item do *Product Backlog*	Tarefa	Estimativa inicial (em horas)
2	Definição da abordagem: tipo de tarefa de *data mining*, algoritmo e *baseline*	18
2	Desenvolvimento da infraestrutura experimental: bases de dados, preparação dos dados, implementação de scripts experimentais	36
2	Modelação dos dados com vista a obter modelos de previsão dos dados de vendas ao nível da categoria	54
2	Afinação dos parâmetros dos algoritmos	30
2	Análise dos resultados	24

Sprint 3

Item do *Product Backlog*	Tarefa	Estimativa inicial (em horas)
3	Definição da abordagem: meta heurística, algoritmo e *baseline*	32
3	Aplicação do algoritmo de otimização	52
3	Afinação dos parâmetros do algoritmo	24
3	Análise dos resultados	24

Depois dos objetivos para o Sprint estarem definidos, qualquer alteração deve ser deferida para o próximo Sprint. Isto significa que se o *Product Owner* a meio do Sprint decidir que existe um novo item que deseja ver implementado, essa alteração não pode ser realizada até ao início do próximo Sprint. Se uma circunstância externa aparecer que signifique que a equipa está a perder tempo caso continue o trabalho, o *Product Owner* ou a equipa podem terminar o Sprint. Neste caso, a equipa acaba o Sprint no momento, e é realizada uma nova reunião de planeamento de forma a iniciar um novo Sprint.

Existe uma influência positiva originada pela equipa estar protegida de alterações aos objetivos durante o Sprint. A equipa trabalha sabendo com certeza absoluta que o objetivo não irá ser alterado, e disciplina o *Product Owne*r a definir cuidadosamente os itens do *Product Backlog*. Seguindo estas regras o *Product Owner* ganha a confiança de saber que a equipa está comprometida a fazer o seu melhor e a completar um conjunto de trabalho realista que escolheu, e sabe que tem a possibilidade de realizar as alterações que pretender ao *Product Backlog* antes de começar o próximo Sprint. Nesse momento, adições, exclusões, modificações e priorizações são possíveis e aceitáveis, eliminando assim o estigma à volta de mudanças como alterações nos requisitos e objetivos.

3.2.4.2 *Scrum* Diário

A reunião de *Scrum* diário tem como objetivo a atualização e coordenação de tarefas entre membros da equipa. Nesta reunião deve participar a equipa e o *Scrum Master*, assegurando que a equipa realiza a reunião. A presença do *Product Owner* é opcional. As reuniões diárias de *Scrum* devem ser curtas, nunca ultrapassando os 15 minutos, e acontecer todos os dias. De forma a manter-se curta, é aconselhado que todos os elementos se mantenham de pé.

Na reunião de *Scrum* é recomendado não existir a presença de gestores ou indivíduos com uma posição de autoridade na organização, pois a sua presença pode fazer a equipa sentir-se monitorizada, sob pressão para reportar progresso diário, e inibindo-se de reportar o que é realmente fundamental, os obstáculos encontrados.

O objetivo destas reuniões diárias é dar aos membros da equipa a oportunidade de sincronizar o seu trabalho e reportar todos os obstáculos encontrados. Na reunião de *Scrum*, um elemento de cada vez, responde a três pontos:

- O que foi atingido desde a última reunião de *Scrum*;
- O que será realizado entre esta reunião e a próxima;
- O que está a impedir o progresso.

No caso de serem reportados obstáculos, um elemento da equipa deve tomar nota do mesmo, sendo o *Scrum Master* responsável por ajudar a equipa a resolvê-los. Na reunião de

Scrum não deve ser aprofundada a discussão, o objetivo é responder aos três pontos. Caso a discussão seja necessária, deve ser tida imediatamente depois da reunião de *Scrum*, numa nova reunião.

A reunião de discussão após a reunião de *Scrum* deve consistir numa reunião onde alguns elementos da equipa, discutem e procuram formas de se adaptarem à informação recolhida da reunião diária de *Scrum*.

Acompanhar o progresso durante o Sprint

Para que a que a equipa consiga realizar a sua própria gestão, é necessário que saiba de que forma está a progredir. Com esse objetivo, os membros da equipa devem atualizar, todos os dias, as suas estimativas do esforço restante para completar as tarefas do Sprint *Backlog*. Deve ser também colocado o esforço restante da equipa como um todo, de forma a poder criar um gráfico de Burndown com esses dados. Este gráfico mostra, a cada dia, uma nova estimativa da quantidade de trabalho restante para a equipa terminar as tarefas do Sprint *Backlog*. Idealmente, é um gráfico decrescente, como uma trajetória que permite atingir o 0 de esforço restante no último dia do Sprint (Figura 13).

O gráfico de *Burndown* deve demonstrar quanto a equipa já progrediu em relação ao objetivo em termos da quantidade de trabalho restante para o Sprint. Caso a linha do gráfico não apresente um comportamento decrescente em direção a 0, perto do final do Sprint, a equipa necessita de se ajustar, como por exemplo reduzir a capacidade por Sprint ou encontrar uma forma de trabalho mais eficaz.

Para o exemplo da Figura 12 foi assumido um Sprint de 4 semanas, com 22 dias de trabalho. A capacidade da equipa para o Sprint foi determinada, utilizando a estimativa de que são realizadas 6 horas de trabalho diário para os itens do Sprint *Backlog*, resultando numa capacidade de 132 horas por Sprint.

A linha a azul do gráfico da Figura 13, representa o total de esforço restante, atingindo o 0 no final do Sprint, o que significa que todo o trabalho planeado para o Sprint foi concluído. Caso isso não se verifique as tarefas não concluídas passavam para o Sprint *Backlog* do Sprint seguinte.

		Estimativas diárias de esforço realizado																				
Tarefas	Estimativas iniciais	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Recolha inicial de dados	16	4	4	4	2	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Limpeza dos dados	20	0	2	2	2	4	4	4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Análise exploratória dos dados	20	0	0	0	2	0	0	2	2	4	4	2	2	2	0	0	0	0	0	0	0	0	0
Seleção dos dados e das variáveis a utilizar no projeto	20	0	0	0	0	0	0	0	0	2	2	2	4	4	4	2	0	0	0	0	0	0	0
Definir indicadores da representatividade dos dados das categorias	24	0	0	0	0	0	0	0	0	0	0	0	0	0	2	4	4	6	4	2	2	0	0
Encontrar categorias com maior volatilidade na área de venda	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	4	6	4	6	8
Total de esforço restante	132	4	6	6	6	6	4	6	4	6	6	4	6	6	6	6	6	8	8	8	6	6	8

Figura 12 : Exemplo de um Sprint *Backlog* com estimativas diárias de trabalho restante

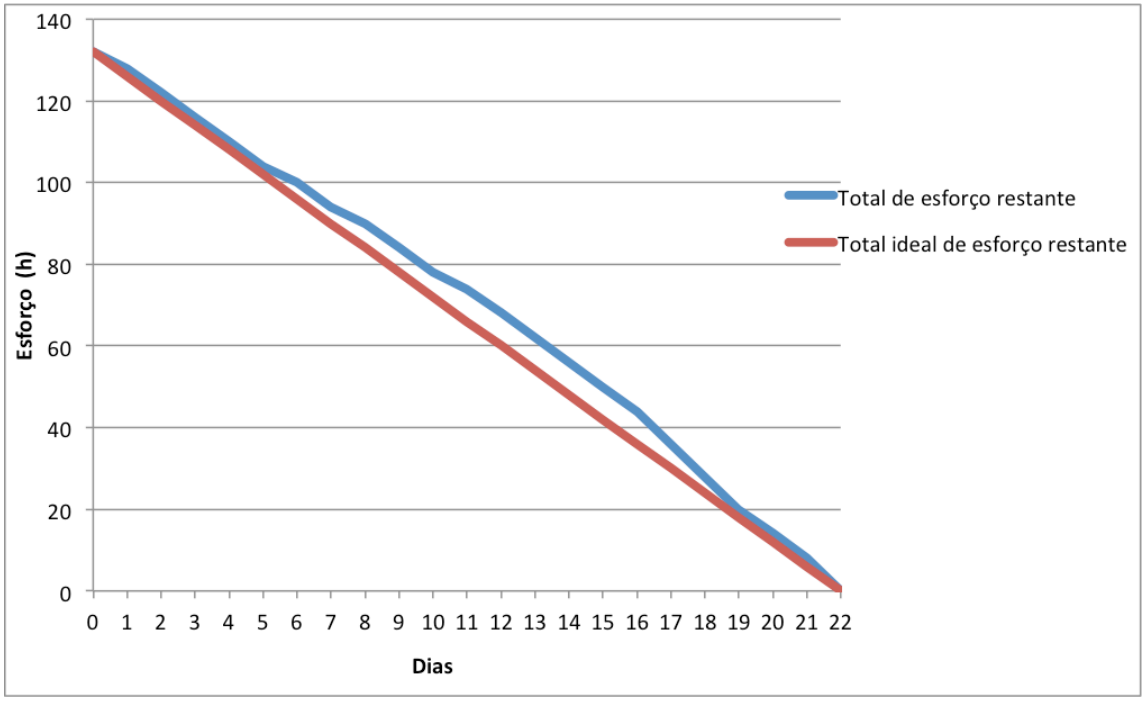

Figura 13 : Gráfico de *Burndown*

3.2.4.3 Refinamento do *Product Backlog*

A reunião de refinamento do *Product Backlog* tem como objetivo analisar, estimar, priorizar e dividir itens de grande dimensão do *Product Backlog*, para Sprints futuros.

Nesta reunião deve participar a equipa, o *Product Owner* caso tenha conhecimento do processo de forma a poder ajudar no refinamento. Caso contrário deve apenas estar presente numa pequena parte da reunião para priorizar os itens. A presença do *ScrumMaster* é aconselhada de forma a ajudar que a equipa seja eficaz. Durante o Sprint, uma percentagem de trabalho deve ser dedicada para a reunião de refinamento do *Product Backlog*, não devendo exceder 10% da capacidade de trabalho da equipa por Sprint.

O refinamento do *Product Backlog* inclui, análise detalhada dos requisitos, divisão de itens de grande dimensão, estimação de novos itens e estimação de itens existentes. Este trabalho de refinamento é apenas realizado para itens a serem desenvolvidos nos próximos Sprints. Um sinal de que esta reunião não está a ser bem sucedida, é a reunião de planeamento do Sprint envolver questões significativas de descoberta ou confusão acerca dos itens do *Product Backlog*.

3.2.4.4 Revisão do Sprint

A reunião de revisão do Sprint tem como objetivo a inspeção e adaptação dos resultados do Sprint. Nesta reunião devem estar presentes a equipa, o *Product Owner*, o *ScrumMaster*, e outros *stakeholders* convidados pelo Product Owner. A duração da reunião não deve exceder 1

hora por cada semana de duração do Sprint. Durante a reunião todos os envolvidos são livres para colocar questões e partilhar a sua opinião. O principal objetivo é inspecionar o resultado do Sprint. É um período para o *Product Owner* analisar o que está a acontecer com o produto e com a equipa, e para a equipa entender as necessidades do *Product Owner* e do mercado. Um elemento crítico da revisão é uma conversa profunda entre a equipa e o *Product Owner* de forma a ambos compreenderem a situação atual do projeto e procurarem conselhos.

A reunião de revisão deve incluir uma demonstração do que foi desenvolvido durante o Sprint. Esta demonstração não deve ser uma apresentação dada pela equipa. Deve ser uma inspeção do resultado pelo *Product Owner*. É preferível uma sessão ativa onde utilizadores reais e o *Product Owner* interajam com o resultado, em vez de uma sessão passiva de demonstração por parte da equipa.

3.2.4.5 Retrospetiva do Sprint

A reunião de retrospetiva do Sprint envolve inspeção e adaptação da equipa em relação ao processo e ambiente de desenvolvimento. Nesta reunião deve participar a equipa, o *Scrum Master* e opcionalmente o *Product Owner*. A retrospetiva deve ser seguida da reunião de revisão, e a sua duração não deve exceder os 45 minutos por semana de duração do Sprint.

É uma oportunidade para a equipa discutir o que está a funcionar e o que não está, no processo, e acordar alterações a tentar.

Algumas equipas focam a reunião retrospetiva apenas nos problemas, e isso é errado. Esta abordagem pode levar a que os elementos da equipa pensem que é uma reunião depressiva ou negativa. Em vez disso, a reunião de retrospetiva deve ser focada também no que correu bem.

Como iniciar o próximo Sprint

Depois da reunião de revisão do Sprint, o *Product Owne*r pode atualizar o *Product Backlog*, sendo responsável por assegurar que as alterações estão refletidas no documento.

Entre Sprints, não existem tempos mortos. Normalmente é realizada a reunião de retrospetiva do Sprint num dia, e no dia de trabalho seguinte é realizada pela manhã a reunião de planeamento do Sprint.

Os Sprints continuam até que o Product Owner decida que o produto está pronto para evoluir para a fase de *Deployment*. A visão perfeita é que no final de cada Sprint o produto possa potencialmente ser entregue ao cliente, já testado e documentado. A implicação é que tudo esteja completamente acabado em cada Sprint, sendo possível no final de cada Sprint realizar o *Deployment* do produto caso o *Product Owner* o pretenda.

Capítulo 4

Estudos de caso

Neste capítulo é descrito um estudo de caso retrospetivo, que demonstra a validade da metodologia Scrum-DM para projetos de *data mining* e dois estudos de caso da aplicação prática da metodologia em projetos.

4.1 Aplicação retrospetiva da metodologia

De forma a validar a metodologia para projetos de *data mining*, foram realizados dois estudos de caso retrospetivos. Um dos casos é utilizado no capítulo anterior de forma a ilustrar com exemplos a metodologia Scrum-DM, o segundo caso é descrito de seguida.

4.1.1 Caso A

O caso A consiste na aplicação da metodologia Scrum-DM a um projeto desenvolvido no INESC TEC com o objetivo de desenvolver um sistema que permita otimizar o desempenho de uma cadeia logística e o desenvolvimento de um sistema de recomendações. Ambos os sistemas estão apoiados em dados recolhidos de etiquetas RFID dos produtos da área têxtil. Neste projeto, os objetivos foram definidos pelo negócio.

A metodologia de *data mining* adotada pela equipa de desenvolvimento foi o CRISP-DM, sendo a equipa constituída por dois elementos. Durante o projeto foram entregues ao cliente vários resultados: um relatório do estado da arte numa fase inicial, os modelos e sua avaliação empírica na fase final de desenvolvimento e no final um sistema para o cliente integrar com o seu sistema.

Durante o desenvolvimento as tarefas foram inicialmente definidas com elevada granularidade pelo cliente, tendo sido posteriormente definidas com sucessiva diminuição da granularidade de forma reativa. Isto é, era definida uma nova tarefa sempre que a anterior era terminada. Para os elementos da equipa de desenvolvimento existiram tarefas a serem executadas em paralelo.

Este projeto teve a duração de cerca de 6 meses, tendo a fase de *Business Understanding* durado cerca de 1 mês, 3 meses para *Data Understanding, Data Preparation e Modeling,* em sobreposição, 1 mês a fase de *Evaluation* e 1 mês a fase de *Deployment*.

Equipa que desenvolveu este projeto era constituída por 4 elementos o que sugere a possibilidade de aplicar a metodologia Scrum-DM neste projeto. Para isso, teriam de ser atribuídos os papéis de *Product Owner* e *Scrum Master*, para além da equipa de desenvolvimento.

Depois dos papéis estarem atribuídos seria necessário a fase inicial de *Business Understanding* da metodologia Scrum-DM para definir o *Product Backlog* para o projeto. Na Tabela 10 é descrito um exemplo do *Product Backlog* para este caso na fase inicial do projeto.

Tabela 10 : *Product Backlog* para o estudo retrospetivo do caso A

Data Mining Stories	Critério de aceitação
Como gestor logístico, quero um sistema de recomendação tendo por base dados RFID de experiência do cliente em loja, de forma a otimizar a gestão do espaço.	Relatório com pelo menos 3 modelos e estimativas do seu desempenho de acordo com métricas aceites na comunidade científica
Como analista, quero analisar as fontes de dados e os dados, de forma a aumentar o conhecimento sobre os mesmos e sobre o problema.	Relatório discutindo as características de todas as fontes de dados disponibilizadas, incluindo estatísticas e gráficos de cada variável e pelo menos 5 pares de variáveis consideradas interessantes pelo analista, bem como a justificação dessa escolha
Como analista, quero realizar a limpeza dos dados de forma a melhorar a sua qualidade.	Conjunto de dados obtido a partir do conjunto original, resolvendo as questões de qualidade avaliadas como mais importantes para o modelo final pelo analista
Como analista, quero cestos de produtos com base nos dados RFID de experiência do cliente em loja, de forma a puder usar algoritmos de recomendação para identificar padrões no seu	Conjunto de dados preparados para serem usado pelos métodos de sistemas de recomendação.

comportamento.	
Como analista, quero um modelo de recomendação tendo por base cestos, de forma a poder alimentar um sistema de recomendação de produtos.	3 modelos de recomendação
Como analista, quero avaliar os modelos de recomendação gerados, de forma a ter uma estimativa do seu desempenho na prática.	Estimativa do desempenho dos modelos de recomendação obtidos, de acordo com métricas aceites pela comunidade

Posteriormente à fase de *Business Understanding*, a equipa, em conjunto com o *Product Owner* e o *Scrum Master* deveriam definir a duração de cada Sprint. Tendo em conta os dados de duração de cada fase do CRISP-DM do projeto, a duração ideal de cada Sprint para este projeto seria de 1 mês. Depois da duração dos Sprints estar decidida, aconteceria a reunião de planeamento do primeiro Sprint, onde o *Product Owner* deveria priorizar os itens do *Product Backlog*. Por seu lado, a equipa seleciona os itens do topo do *Product Backlog* priorizado, até preencher a sua capacidade de trabalho para o Sprint, decompondo posteriormente o trabalho em tarefas tendo em conta os itens selecionados do *Product Backlog* e as fases do CRISP-DM necessárias para as desenvolver. Para cada Sprint as tarefas da reunião de planeamento do Sprint devem ser repetidas.

Na Tabela 11 está representado um exemplo do Sprint *Backlog* para 5 Sprints deste projeto. Em cada Sprint é selecionado um item do *Product Backlog* e são definidas as tarefas necessárias para o desenvolver. Por ser uma aplicação retrospetiva da metodologia, isto é, uma aplicação posterior ao projeto estar concluído não foi estimado o esforço dos itens do *Product Backlog* nem das tarefas para cada Sprint devido a falta de dados, embora essa atividade fosse imprescindível numa aplicação real da metodologia.

Tabela 11 : Sprint *Backlog* para o estudo retrospetivo do caso A

Primeiro Sprint Itens do *Product Backlog* selecionados: Como analista, quero analisar as fontes de dados e os dados, de forma a aumentar o conhecimento sobre os mesmos e sobre o problema
Tarefas:
Análise do dicionário de dados
Análise exploratória dos dados
Identificar e documentar problemas nos dados
Segundo Sprint

Itens do *Product Backlog* selecionados: Como analista, quero realizar a limpeza dos dados de forma a melhorar a sua qualidade
Tarefas:
Definir estratégias de melhoria de qualidade dos dados
Desenvolver funções de limpeza dos dados
Limpeza de erros nos dados
Terceiro Sprint Itens do *Product Backlog* selecionados: Como analista, quero realizar a limpeza dos dados de forma a melhorar a sua qualidade
Tarefas:
Definição do critério de agrupamento das transações em cestos
Desenvolver funções de criação de cestos
Avaliação das características de cestos criados com vários parâmetros
Quarto Sprint Itens do *Product Backlog* selecionados: Como analista, quero um modelo de recomendação tendo por base cestos, de forma a poder alimentar um sistema de recomendação de produtos
Tarefas:
Seleção do algoritmo de recomendação a usar a partir das alternativas disponíveis
Desenvolver funções de geração de modelos
Geração dos modelos
Documentar modelos gerados
Quinto Sprint Itens do *Product Backlog* selecionados: Como analista, quero avaliar os modelos de recomendação gerados, de forma a ter uma estimativa do seu desempenho na prática
Definição da metodologia de avaliação
Desenvolver funções de avaliação de modelos
Avaliação experimental dos modelos
Documentar resultados da avaliação experimental

4.1.1.1 Conclusões

O estudo deste caso de uma aplicação retrospetiva da metodologia permite confirmar a validade da metodologia para projetos de data mining. Podemos concluir que é possível após uma fase de *Business Understanding* normal do CRISP-DM converter os objetivos identificados em *Data Mining Stories* e associar cada uma a um critério de aceitação. Podemos também concluir que é possível a utilização do CRISP-DM como guia para a definição das tarefas do Sprint *Backlog*, tendo como objetivo o desenvolvimento de uma *Data Mining Story*, respeitando o critério de aceitação.

4.2 Aplicação da metodologia num projeto de *data mining*

De forma a avaliar a aplicabilidade da metodologia em projetos de *data mining*, a mesma foi aplicada num projeto real. Este projeto tem como principal propósito o desenvolvimento de ferramentas que permitam incidir um estudo pormenorizado de registos provenientes de chamadas telefónicas de uma operadora móvel. Assim, o principal enfoque passará pela implementação e avaliação de algoritmos de agrupamento de dados, para além da implementação e avaliação de algoritmos de deteção de anomalias em tempo real.

O objetivo do negócio, é a redução de custos provenientes de fraude de utilizador, com especial relevância para fraude por superimposição. Um exemplo típico de fraude por superimposição pode ser visto quando um utilizador com um plano de tráfego nacional de repente tem um súbito aumento de chamadas internacionais. Este cliente poderá estar a ser alvo de clonagem do cartão.

Finalmente torna-se importante definir o conceito de fraude, também conhecida por anomalia, *outlier*, desvio ou simplesmente erro. Segundo Hawkins (Hawkins, 1980), uma anomalia ou *outlier,* é uma observação que se desvia tanto das restantes de forma a criar suspeitas de que foi gerada por um mecanismo diferente.

Neste projeto, os dados dos utilizadores são provenientes do registo operacional de chamadas telefónicas de uma operadora nacional. Estes registos têm uma janela de ocorrências que perdura por um período de aproximadamente seis meses. A média de registos diários é da ordem dos 8 milhões. Assim, a plataforma desenvolvida deverá ser capaz de trabalhar com um fluxo contínuo de dados que pode alcançar os 93 registos por segundo.

Cada registo é caracterizado pelo seguintes atributos fixos:

- Identificadores dos interlocutores da chamada e dos seus contratos;
- Locais de emissão e receção da chamada;
- Marca temporal de início da chamada;

- Duração da chamada;
- Direção da chamada;
- Tipo de destino (local ou internacional);
- Tipo da chamada;
- Operador;
- *Voicemail* (Sim ou não);
- Chamada perdida (Sim ou não).

4.2.1 Planeamento

Para a aplicação da metodologia no projeto foram definidos os papéis necessários, *Product Owner*, *ScrumMaster* e equipa de desenvolvimento, tendo o autor deste trabalho ficado com o papel de *ScrumMaster* de forma a garantir a correta aplicação da metodologia. A equipa de desenvolvimento é constituída por dois elementos.

Depois dos papéis estarem definidos foi realizada a fase de *Business Understanding* como descrita na metodologia Scrum-DM na Secção 4.1.2.1. No final desta fase ficou definido um épico para o projeto e as *Data Mining Stories* para um mês de trabalho e os seus critérios de aceitação (Tabela 12).

Tabela 12 : Conjunto de *Data Mining Stories* e critérios de aceitação para o projeto

Épico	Critério de aceitação
Como gestor operacional, quero uma plataforma de detecção de anomalias em chamadas telefónicas de forma a reduzir os custos com as anomalias.	Plataforma de deteção de anomalias com resultados para 1 dia de registos

Data Mining Stories	Critério de aceitação
Como analista, quero avaliar empiricamente os algoritmos de *clustering* disponíveis de forma a selecionar o mais adequado para a criação de perfis de clientes	Comparação empírica de pelo menos 3 algoritmos de *clustering*

Como analista, quero criar variáveis que permitam definir o perfil de clientes e avaliar os *clusters* resultantes	Conjunto de variáveis para definição de perfis em tempo útil, com comparação de desempenho respetivo
Como analista, quero ter disponíveis todos os resultados dos *clusterings* já realizados de forma a perceber a sua evolução	Cinco *clusters* guardados na base de dados
Como analista, quero ter um método de detecção de anomalias com base num algoritmo de *clustering* incremental de forma a detetar chamadas anómalas	Resultados da detecção de anomalias para um dia de chamadas com pelo menos uma chamada anómala detetada
Como analista, quero avaliação de bibliotecas de visualização em R com interligação ao *Shiny*, de forma a selecionar uma para desenvolver uma ferramenta de visualização de chamadas	Tabela que compare pelo menos 3 bibliotecas de acordo com pelo menos 5 critérios
Como analista, quero uma ferramenta de visualização de chamadas de forma a aumentar o conhecimento da base de dados dos registos.	Protótipo da interface de visualização de chamadas
Como analista, quero definir um conjunto de estatísticas robustas de avaliação de chamadas, de forma a caracterizar os registos anómalos.	Relatório com a definição e evolução de resultados das diferentes estatísticas ao longo de um mês de dados.
Como analista, quero definir, modelar e comparar diferentes mecanismos de avaliação de falhas de forma a incorporar os registos com os algoritmos de *clustering*.	Relatório com os diferentes resultados, tendo em conta um mês de registos.

4.2.2 Sprints e Reuniões

O *Product Backlog* para o projeto foi definido na primeira reunião de planeamento do Sprint, onde as Data Mining Stories foram priorizadas pelo *Product Owner* e a equipa estimou o esforço necessário para cada item (Tabela 13). O *Product Backlog* foi definido para 1 mês de trabalho, sendo também definida a duração de uma semana por Sprint, sendo o primeiro Sprint de 6 dias e o quarto de 4 dias de forma a acertar as semanas com o calendário. A reunião de Planeamento do Sprint teve a duração de cerca de três horas, sendo uma hora para priorização

do *Product Backlog*, e as restantes duas horas para a definição do Sprint *Backlog* pela equipa, para o primeiro Sprint. Devido ao tamanho reduzido dos Sprints, as reuniões de refinamento do *Product Backlog* foram incorporadas nas reuniões de planeamento do Sprint.

O *Product Backlog* (Tabela 13) foi priorizado pelo Product Owner e estimado pela equipa.

Tabela 13 : Product Backlog para o projeto

Prioridade	*Data mining Story*	Critério de aceitação	Estimativas iniciais de esforço
1	Como analista, quero avaliar empiricamente os algoritmos de *clustering* disponíveis de forma a selecionar o mais adequado para a criação de perfis de clientes	Comparação empírica de pelo menos 3 algoritmos de *clustering*	48 horas
2	Como analista, quero avaliação de bibliotecas de visualização em R com interligação ao *Shiny*, de forma a selecionar uma para desenvolver uma ferramenta de visualização de chamadas	Tabela que compare pelo menos 3 bibliotecas de acordo com pelo menos 5 critérios	48 horas
3	Como analista, quero criar variáveis que permitam definir o perfil de clientes e avaliar os *clusters* resultantes	Conjunto de variáveis para definição de perfis em tempo útil, com comparação de desempenho	40 horas
4	Como analista, quero uma ferramenta de visualização de chamadas de forma a aumentar o conhecimento da base de dados dos registos.	Protótipo da interface de visualização de chamadas	40 horas
5	Como analista, quero ter disponíveis todos os resultados dos *clusterings* já realizados de forma a perceber a sua evolução	Cinco *clusters* guardados na base de dados	40 horas
6	Como analista, quero definir um conjunto de estatísticas robustas de avaliação de	Relatório com a definição e evolução de resultados das diferentes	40 horas

	chamadas, de forma a caracterizar os registos anómalos.	estatísticas ao longo de um mês de dados.	
7	Como analista, quero ter um método de detecção de anomalias com base num algoritmo de *clustering* incremental de forma a detetar chamadas anómalas	Resultados da detecção de anomalias para um dia de chamadas com pelo menos uma chamada anómala detetada	30 horas
8	Como analista, quero definir, modelar e comparar diferentes mecanismos de avaliação de falhas de forma a incorporar os registos com os algoritmos de *clustering*.	Relatório com os diferentes resultados, tendo em conta um mês de registos.	30 horas

Durante a reunião de planeamento do Sprint, depois do *Product Owner* ter priorizado os itens do *Product Backlog* e a equipa ter estimado o esforço para cada item, a equipa selecionou da lista os itens para desenvolver no Sprint, tendo sido selecionados o número 1 e 2 do *Product Backlog*. Depois desta seleção estar efetuada a equipa decompôs os dois itens em tarefas, desenvolvendo o Sprint *Backlog* para o primeiro Sprint.

De forma a acompanhar o progresso durante todo o projeto, existiram reuniões diárias de Scrum, com a duração de cerca de 10 minutos, onde foram discutidos os pontos referidos na Secção 4.1.4.2, e atualizado o Sprint *Backlog* (Tabela 14) com as estimativas de esforço diário realizado por tarefa e com esses dados criado um gráfico de *Burndown* (Figura 14).

Para o primeiro Sprint a capacidade de trabalho da equipa foi estimada em 96 horas, isto é, 2 elementos da equipa, 8 horas diárias de trabalho para o Sprint durante 6 dias.

Tabela 14 : Sprint *Backlog* para o primeiro Sprint

		Estimativa de esforço diário realizado					
Tarefa	Esforço estimado	1	2	3	4	5	6
Identificar e instalar as diferentes opções (tarefa concluída)	10	3	0	4	0	3	0
Testar (tarefa concluída)	30	5	2	2	5	8	8
Avaliar (tarefa concluída)	7	1	1	1	0	4	0
Reunião para seleção (tarefa concluída)	1	0	0	0	0	0	1
Colocar algoritmos FP/PG no GUI do	10	4	2	0	0	4	0

MOA (tarefa concluída)							
Análise e adaptação dos algoritmos à interface *Instance* do MOA (tarefa concluída)	20	0	4	8	2	2	4
Integração com o *Instance Generator* do MOA (tarefa concluída)	18	0	0	2	8	4	4
Total de esforço realizado por dia pela equipa		13	9	17	15	25	17

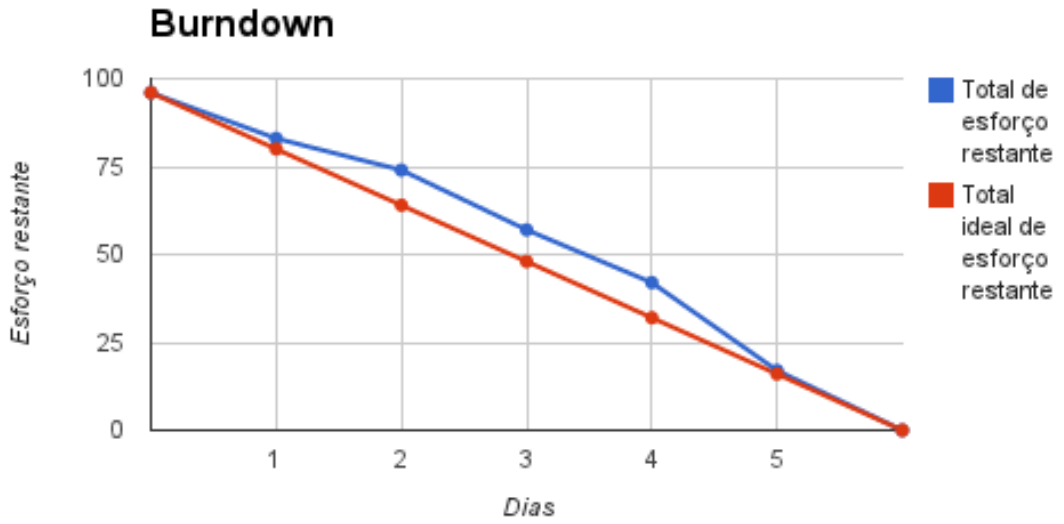

Figura 14 : Gráfico de *Burndown* para o primeiro Sprint

Embora o primeiro Sprint tenha sido de adaptação da equipa à nova metodologia, o Sprint correu bem, tendo todas as tarefas planeadas sido finalizadas. A equipa concluiu os dois itens do *Product Backlog* selecionados, respeitando os seus critérios de aceitação.

No final do primeiro Sprint a equipa realizou a reunião de revisão, onde os resultados do Sprint foram demonstrados ao *Product Owner*, tendo o mesmo aprovado o que foi realizado no Sprint. Esta reunião teve a duração de cerca de 30 minutos, e estiveram presentes a equipa, o *Product Owner* e o *Scrum Master*.

Depois da reunião de revisão a equipa teve a reunião de retrospetiva do Sprint em conjunto com o *Scrum Master*. Nesta reunião foi relatado pela equipa alguma dificuldade na definição e estimação das tarefas. A reunião de retrospetiva teve a duração de cerca de 15 minutos.

No dia de trabalho seguinte a realizar a reunião de retrospetiva do primeiro Sprint, foi realizada a reunião de planeamento do segundo Sprint e o mesmo foi iniciado. Esta reunião teve a duração de cerca de 2 horas. No final da primeira parte da reunião de planeamento do segundo

Sprint, ficou definido o *Product Backlog* atual, priorizado pelo *Product Owner* e os itens estimados pela equipa (Anexo D).

Na segunda parte da reunião de planeamento do segundo Sprint, a equipa selecionou os dois itens com maior prioridade do *Product Backlog* para desenvolver no Sprint. Depois desta escolha ser efetuada a equipa decompôs os itens em tarefas (Tabela 15), e calculou o esforço para cada.

Para o segundo Sprint a capacidade de trabalho da equipa foi estimada em 80 horas, menos 16 horas que para o primeiro Sprint devido a este Sprint ter 5 dias, ao contrário dos 6 do primeiro.

Tabela 15 : Sprint *Backlog* para o segundo Sprint

		Estimativa de esforço diário realizado				
Tarefa	Esforço estimado	1	2	3	4	5
Definição do modelo da base de dados (tarefa não concluída)	16	2	1	3	2	4
Definição do modelo do visualizador de chamadas, e identificação de principais problemas (tarefa não concluída)	16	2	3	0	3	0
Conexão com API de visualização (tarefa não concluída)	8	1	2	1	1	0
Exportar dados para base de dados *Sqlite* (tarefa concluída)	10	6	4	0	0	0
Desenvolvimento de *middleware* para interface com a base de dados (tarefa concluída)	12	0	0	6	6	0
Criação de motor de simulação direto da base de dados (tarefa concluída)	6	0	0	0	3	3
Criar variáveis simples para caracterizar perfis de utilizadores (tarefa não concluída)	6	0	0	0	0	3
Avaliação dos perfil simples segundo um algoritmo de *clustering* (tarefa não concluída)	6	0	0	0	0	3
Total de esforço realizado por dia pela equipa		11	10	10	15	10

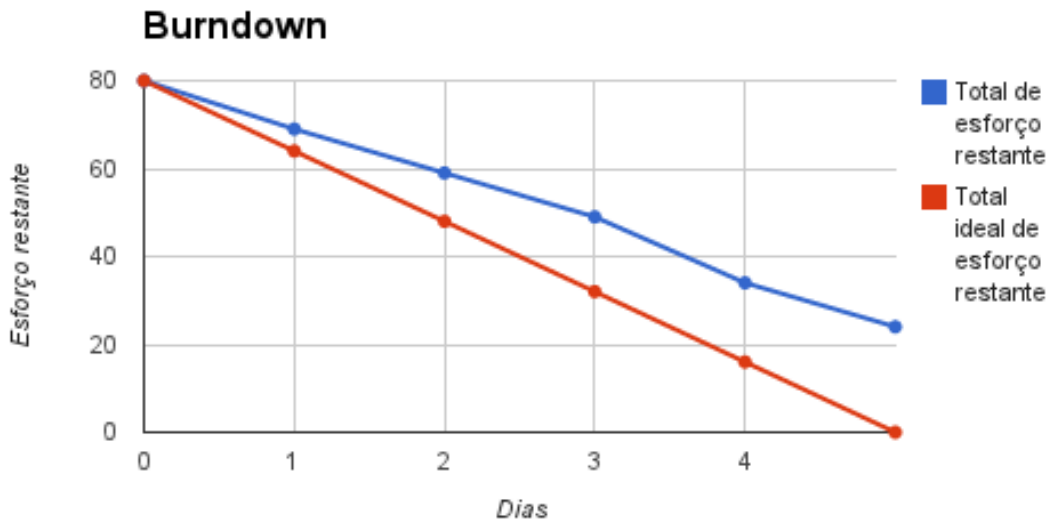

Figura 15 : Gráfico de *Burndown* para o segundo Sprint

No segundo Sprint a equipa subestimou o esforço necessário para desenvolver os itens do *Product Backlog*, e o esforço para concluir as tarefas planeadas. Por esta razão, quatro das tarefas planeadas ficaram por concluir (Figura 15), passando para o próximo Sprint, não tendo sido fechado qualquer item do *Product Backlog*.

Na reunião de revisão do segundo Sprint a equipa demonstrou ao *Product Owner* os resultados das tarefas que ficaram terminadas no Sprint, e quais os impedimentos encontrados para as restantes não terem sido concluídas. Esta reunião teve uma duração de aproximadamente 15 minutos.

Na reunião de retrospetiva foi analisado o facto da equipa não ter cumprido as tarefas planeadas para o Sprint, tendo-se acordado na redução da capacidade da equipa para os Sprints seguintes, de 8 horas de trabalho diário por elemento da equipa, para tarefas relacionadas com o Sprint, para 6 horas. Foi também discutida a necessidade de diminuir a granularidade das tarefas planeadas, de forma a aumentar o grau de certeza das estimativas. A reunião de retrospectiva teve uma duração de 20 minutos.

No dia seguinte à reunião de retrospetiva, a equipa, em conjunto com o *Product Owner* e o *Scrum Master*, realizou a reunião de planeamento do terceiro Sprint. Esta reunião durou aproximadamente 1 hora e meia. No final da primeira parte da reunião ficou definido o *Product Backlog* atual, priorizado pelo *Product Owner* e estimado pela equipa (Anexo D).

Na segunda parte da reunião de planeamento a equipa selecionou os três itens com maior prioridade do *Product Backlog*. Depois da seleção dos itens ter sido efetuada, a equipa decompôs os itens em tarefas e estimou o esforço para cada (Tabela 16).

Para o terceiro Sprint a capacidade de trabalho da equipa foi reduzida para as 60 horas, isto é, 6 horas de trabalho diário por elemento da equipa, durante 5 dias.

Tabela 16 : Sprint *Backlog* para o terceiro Sprint

		Estimativa de esforço diário realizado				
Tarefa	Esforço estimado	1	2	3	4	5
Definição do modelo da base de dados (*sqlite* / *sqlserver* no R) (tarefa concluída)	4	1	3	0	0	0
Definição do modelo do visualizador de chamadas, e identificação de principais problemas (tarefa concluída)	8	2	0	4	0	2
Conexão com API de visualização (tarefa concluída)	4	0	0	3	1	0
Definição e implementação do mecanismo animado (tarefa concluída)	14	1	6	5	2	0
Criar variáveis simples para caracterizar perfis de utilizadores (tarefa não concluída)	6	4	0	0	0	0
Avaliação dos perfis segundo um algoritmo de *clustering* (tarefa não concluída)	6	0	0	0	4	0
Guardar resultados de *clustering* em bases de dados (tarefa concluída)	6	0	0	6	0	0
Filtrar *stream* de eventos anómalos (tarefa concluída)	6	0	0	0	0	6
Iniciar protótipo de visualização em D3.js (tarefa concluída)	6	0	6	0	0	0
Total de esforço realizado por dia pela equipa		12	9	12	7	8

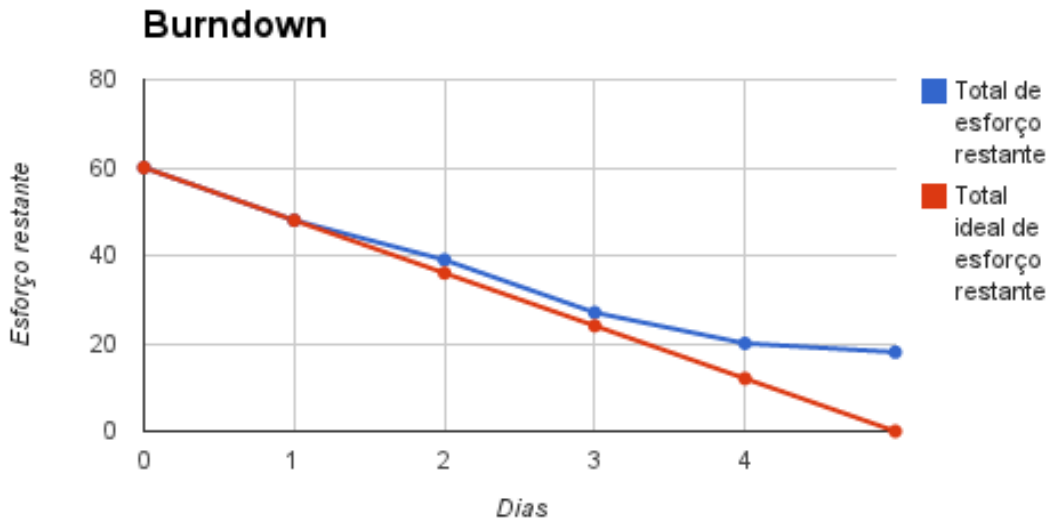

Figura 16 : Gráfico de *Burndown* para o terceiro Sprint

No terceiro Sprint embora não tenham ficado concluídas todas as tarefas planeadas para o Sprint (Figura 16), ficaram concluídos os dois primeiros itens do *Product Backlog*, dos três selecionados para o Sprint. As tarefas do terceiro item não foram concluídas neste Sprint devido a serem tarefas que necessitaram de ser continuadas no Sprint seguinte de forma a apoiar o desenvolvimento do item com prioridade 5 do *Product Backlog*.

No último dia do terceiro Sprint foi realizada a reunião de revisão do Sprint. Nesta reunião a equipa demonstrou ao *Product Owner* os resultados dos dois itens com maior prioridade do *Product Owner*, concluídos no Sprint, e apresentadas os impedimentos encontrados para não ter sido concluído o terceiro. Esta reunião durou aproximadamente 20 minutos.

Na reunião de retrospetiva do Sprint, que seguiu a reunião de revisão, a equipa em conjunto com o *Scrum Master* discutiu a melhor forma de planear tarefas que tenham de ser continuadas num próximo Sprint, tendo sido acordado que devem ser decompostas de forma a poderem ser planeadas para cada Sprint. Esta reunião teve uma duração de 30 minutos.

No dia de trabalho posterior à reunião de retrospetiva do terceiro Sprint, foi realizada a reunião de planeamento do quarto Sprint. Nesta reunião estiveram presentes o *Product Owner*, a equipa e o *Scrum Master*, tendo durado aproximadamente 1 hora. No final da primeira parte desta reunião ficou definido o *Product Backlog* atual, adicionada uma nova *Data Mining Story*, e estimado pela equipa (Anexo D).

Na segunda parte da reunião de planeamento, a equipa selecionou os quatro primeiros itens do *Product Backlog* para desenvolver no Sprint. Decompondo cada item em tarefas e estimando o esforço necessário para cada (Tabela 17).

Para o quarto Sprint a capacidade de trabalho da equipa foi reduzida para as 48 horas, isto é, 6 horas de trabalho diário por elemento da equipa, durante 4 dias.

Tabela 17 : Sprint *Backlog* para o quatro Sprint

		Estimativa de esforço diário realizado			
Tarefa	Esforço estimado	1	2	3	4
Definição de variáveis a serem geradas no R (falhas/sucessos por tempo, duração por tempo) (tarefa não concluída)	3	1	0	0	0
Aplicação de filtros aos dados (tarefa concluída)	2	0	2	0	0
Identificação de intercepções (tarefa concluída)	7	0	0	7	0
Identificação de duplicados (tarefa concluída)	1	0	1	0	0
Integração com o visualizador de chamadas (tarefa concluída)	10	2	3	1	4
Reunião de aceitação (tarefa concluída)	1	1	0	0	0
Desenvolver variáveis simples para caracterizar perfis de utilizadores (tarefa concluída)	3	0	0	0	3
Avaliação dos perfis simples segundo um algoritmo de *clustering* (tarefa concluída)	3	0	2	0	1
Utilizar *clustering* para detecção de anomalias (tarefa concluída)	9	2	3	4	0
Evolução do protótipo de visualização em D3.js (tarefa concluída)	9	4	1	2	2
Total de esforço realizado por dia pela equipa		10	12	14	10

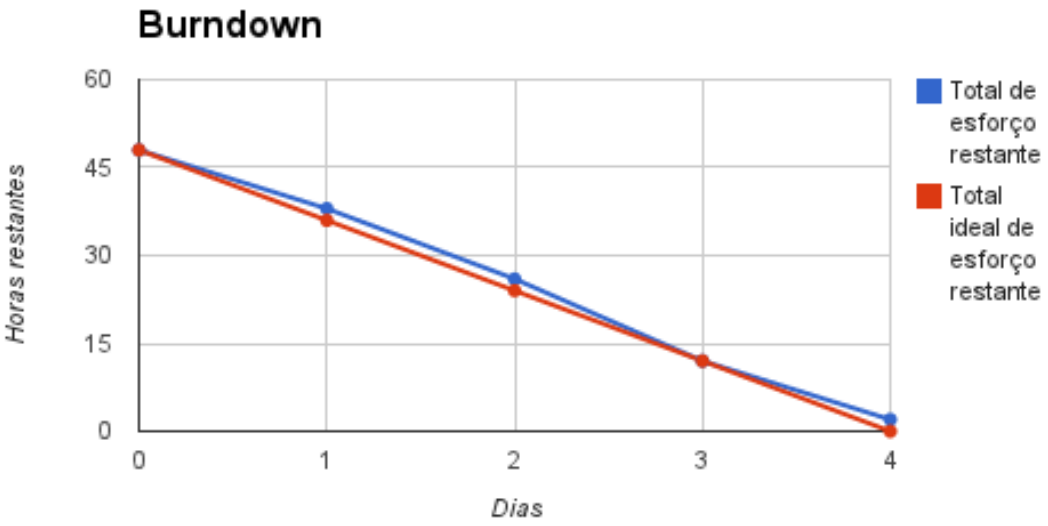

Figura 17 : Gráfico de *Burndown* para o quarto Sprint

No quarto Sprint, existiu subestimação do esforço necessário para uma das tarefas planeadas, tendo ficado por concluir o quarto item do *Product Backlog* (Figura 17). Os três itens com maior prioridade para o Sprint ficaram concluídos.

No final do Sprint realizou-se a reunião de revisão. Nesta reunião foi demonstrado ao *Product Owner* o resultado da conclusão dos três itens do *Product Backlog*, tendo durado aproximadamente 20 minutos.

Posteriormente à reunião de revisão, a equipa, em conjunto com o *Scrum Master* teve a reunião de retrospetiva do Sprint onde se discutiu as razões para existir subestimação do esforço de algumas tarefas, chegando-se à conclusão que é necessário no momento em que se está a definir o Sprint *Backlog*, um maior cuidado a estimar e reduzir a granularidade das tarefas de forma a se conseguir estimativas mais corretas.

Como no final do quatro Sprint é claro um atraso em relação ao que foi planeado no inicio do projeto, quando se definiu o *Product Backlog* para um mês de trabalho, será necessário um novo Sprint, o quinto, de forma a concluir todos os itens do *Product Backlog*. Este atraso deveu-se essencialmente a uma subestimação inicial pela equipa, do esforço necessário para concluir cada item do *Product Backlog* no inicio do projeto.

No dia de trabalho seguinte à reunião de retrospetiva, a equipa realizou, em conjunto com o *Scrum Master* e o *Product Owner* a reunião de planeamento do quinto Sprint. Esta reunião durou aproximadamente uma hora. No final da primeira parte da reunião ficou definido o *Product Backlog* atual, priorizado pelo *Product Owner* e estimado pela equipa (Anexo D).

Na segunda parte da reunião de planeamento, a equipa selecionou os dois itens restantes no *Product Backlog* para desenvolver no Sprint. Decompondo cada item em tarefas e estimando o esforço necessário para cada (Tabela 18).

No quinto Sprint a capacidade de trabalho da equipa foi de 60 horas, 6 horas de trabalho diário por elemento da equipa, durante 5 dias.

Tabela 18 : Sprint *Backlog* para o quinto Sprint

		Estimativa de esforço diário realizado				
Tarefa	Esforço estimado	1	2	3	4	5
Número de falhas por utilizador	7	2	2	3	0	0
Número de sucessos por utilizador	7	2	3	2	0	0
Número de chamadas por utilizador	7	0	0	0	3	4
Duração de chamadas por utilizador	7	0	0	0	3	4
Estatísticas de utilizador (definição de sistema base de cálculo de estatísticas por utilizador)	2	1	0	0	1	0
Visualização de *clustering*	6	6	0	0	0	0
Cálculo da diferença entre dois *clusters*	12	0	6	0	6	0
Visualização da diferença de fois *clusters*	12	0	0	6	0	6
Total de esforço realizado por dia pela equipa		11	11	11	13	14

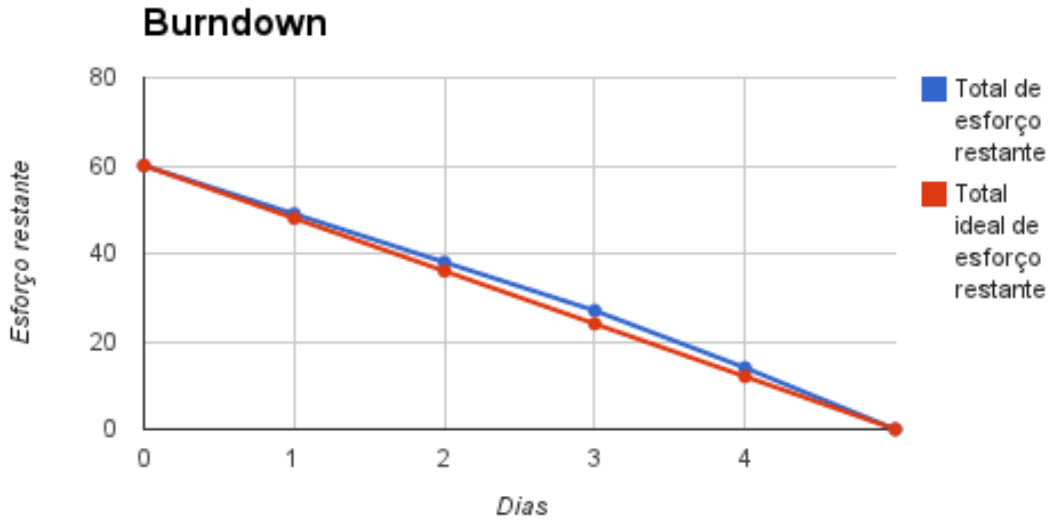

Figura 18 : Gráfico de Burndown para quinto Sprint

No quinto Sprint, todas as tarefas planeadas foram terminadas (Figura 18), tendo ficado concluídos todos os itens do *Product Backlog*.

No final do quinto Sprint, tal como nos anteriores, realizou-se a reunião de revisão do Sprint. Nesta reunião foi demonstrado ao *Product Owner*, o resultado da conclusão dos dois itens do *Product Backlog* selecionados para o Sprint. Esta reunião teve uma duração de aproximadamente 15 minutos.

Posteriormente à reunião de revisão, realizou-se a reunião de retrospetiva do Sprint. Nesta reunião esteve presente a equipa e o *Scrum Master*, e foi analisado o que correu bem e o que correu mal no último Sprint, tendo-se constatado que existiu uma melhoria significativa nas estimativas da equipa em relação à realidade, principalmente por ter existido um maior cuidado na definição das tarefas.

4.2.3 Conclusões

Com este estudo da aplicação da metodologia neste projeto confirma-se que a mesma é aplicável a projetos de *data mining*, em que a equipa de desenvolvimento seja de pelo menos dois elementos.

Por outro lado, verificou-se que a duração planeada para o projeto foi excedida uma semana, o que demonstra que a metodologia dificulta a previsão do tempo necessário para a conclusão de um projeto, principalmente numa fase inicial da adaptação da equipa à metodologia, onde não existem dados históricos da velocidade de trabalho por Sprint.

No final do quinto Sprint foi realizada uma entrevista à equipa de desenvolvimento e ao *Product Owner* de forma a determinar a aceitação à metodologia proposta, e quais as principais dificuldades e impedimentos da adaptação.

Das entrevistas realizadas é possível concluir que a equipa considera que a metodologia permite um desenvolvimento mais rápido em torno de um produto final, permitindo especificar um conjunto de necessidades de implementação em torno do produto. Uma característica importante é que promoveu que os objetivos do projeto fossem pensados com maior profundidade, e análise da diferença entre o que era necessário realizar e aquilo que, de facto, foi possível. Desta forma, tornou mais fácil constatar atrasos em relação ao ideal, enquanto as tarefas ainda estavam em execução.

Foi realçado que a equipa se ajustou dentro do possível à metodologia, embora o domínio da metodologia só acontecerá depois da equipa realizar vários projetos com a mesma metodologia.

Dos conceitos da metodologia, conclui-se que a equipa considera as *Data Mining Stories* como uma boa forma de alinhar os objetivos e a linguagem do negócio com a linguagem técnica

da equipa de desenvolvimento. Concluiu-se também que os critérios de aceitação se demonstram úteis para definir o alvo da equipa de desenvolvimento, realçando a necessidade de entregar um resultado ao cliente no final do Sprint. Adicionalmente, foi observado que o *Product Backlog* e o Sprint *Backlog* permitem que o foco de desenvolvimento seja em terminar os objetivos para o Sprint, permitindo também comunicar as tarefas em processo de execução a todos os intervenientes do projeto.

Do conceito de estimação do esforço, a equipa considera que promove introspeção sobre as tarefas a realizar, embora considere que a estimação pode facilmente estar incorreta, dependendo da familiaridade com as tarefas em questão. Foi também referido que a metodologia obriga a reservar tempo para tarefas que poderão ser contínuas e transversais a todos os Sprints. Estas tarefas pela sua natureza são mais difíceis de estimar que as outras. O aumento da precisão das estimativas através do aumento da granularidade dos objetivos de cada tarefa nem sempre é possível. Isto acontece por exemplo, quando a própria tarefa implica alguma investigação para a sua conclusão, o que torna a tarefa muito difícil de estimar.

4.3 Aplicação da metodologia numa competição do *Data Intelligence Group*

A metodologia Scrum-DM foi aplicada numa competição de *data* mining por uma equipa do *Data Intelligence Group* (DIG*)* da Faculdade de Engenharia da Universidade do Porto.

A competição teve uma duração de 4 semanas e tinha como objetivo prever, partindo de um conjunto de dados históricos, quais os clientes de uma loja que têm maior probabilidade de repetir uma compra depois de receberem um cupão, de forma a otimizar a ação de uma campanha de descontos a serem atribuídos aos clientes.

Para a aplicação da metodologia foram planeados quatro Sprints, cada um com uma duração de uma semana. No final da competição foram realizadas entrevistas à equipa de forma a verificar a aceitação à metodologia proposta, e quais as dificuldades e impedimentos sentidos na adaptação.

4.3.1 Planeamento

Para a aplicação da metodologia no projeto foram definidos os papéis necessários, *Product Owner*, *Scrum Master* e equipa de desenvolvimento, tendo o autor deste trabalho ficado com o papel de *Scrum Master*. A equipa de desenvolvimento é constituída por dois elementos.

Depois dos papéis estarem definidos foi realizada a fase de *Business Understanding da metodologia Scrum-DM,* tendo no final ficado definido o *Product Backlog* para o projeto (Tabela 19).

Tabela 19 : *Product Backlog* para a competição do DIG

Nº	*Data Mining* Stories	Critério de aceitação	Estimativa inicial de esforço
1	Como analista, quero extrair e preparar os dados de forma a poder analisá-los.	Conjunto de dados com pelo menos uma semana de transações.	10h
2	Como analista, quero realizar uma análise preliminar dos dados de forma a aumentar o conhecimento sobre os mesmos.	Conjunto de pelo menos 10 gráficos com caracterização dos dados e enumeração dos problemas de dados identificados.	10h
3	Como analista, quero modelar a relação entre as características dos cupões promocionais e a sua utilização numa compra na loja, de forma a poder decidir se é elegível para utilização numa campanha promocional.	Estimativas de desempenho de modelos obtidos com pelo menos 3 algoritmos diferentes.	10h

4.3.2 Sprints e Reuniões

Na reunião de planeamento do primeiro Sprint a equipa selecionou do *Product Backlog* (Tabela 19) o primeiro item para desenvolver no Sprint. Depois desta seleção ter sido efetuada, a equipa decompôs o item em tarefas criando o *Sprint Backlog* (Tabela 20).

Todos os dias a equipa atualizou o *Sprint Backlog* com as estimativas de esforço diário realizado por tarefa e com esses dados foi criado um gráfico de *Burndown*. Para todos os Sprints a capacidade de trabalho por equipa foi fixada em 10 horas, isto é, 1 hora diária por elemento da equipa. Foi fixado o limite diário de trabalho por elemento da equipa em uma hora por este projeto se tratar de uma competição e as pessoas envolvidas não disporem de muito tempo livre para participar.

Tabela 20 : Sprint *Backlog* para o primeiro Sprint

		Estimativa de esforço diário realizado				
Tarefa	Esforço estimado	1	2	3	4	5

Extrair dados	2	2	0	0	0	0
Criar repositório de dados	8	0	1	1	1	1
Total de esforço realizado por dia pela equipa		2	1	1	1	1

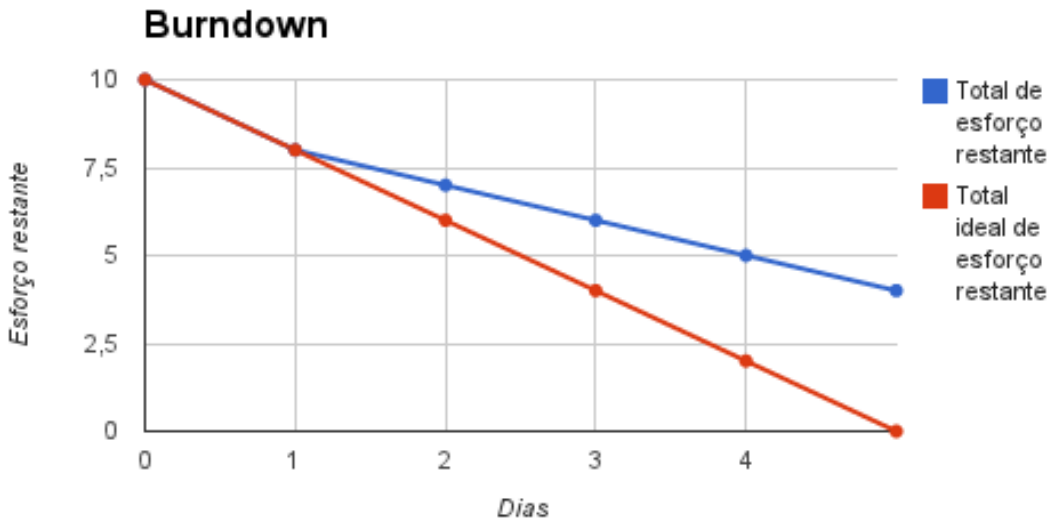

Figura 19 : Gráfico de *Burndown* para o primeiro Sprint

No final do primeiro Sprint o item do *Product Backlog* selecionado para o Sprint não foi terminado (Figura 19) por dificuldade da equipa na criação do repositório de dados. Este impedimento foi reportado na reunião de revisão do Sprint ao *Product Owner* e encontrada uma forma de o solucionar na reunião de retrospetiva do Sprint.

Para o segundo Sprint foi selecionado o primeiro e o segundo item do *Product Backlog* (Tabela 19). Tendo sido decompostos em tarefas (Tabela 21) durante a reunião de planeamento do Sprint.

Tabela 21 : Sprint *Backlog* para o segundo Sprint

		Estimativa de esforço diário realizado			
Tarefa	Esforço estimado	1	2	3	4
Criar repositório de dados	4	1	1	1	1
Análise exploratória	2	0	0	0	1
Limpeza de dados	2	0	0	0	0

Total de esforço realizado por dia pela equipa		1	1	1	2

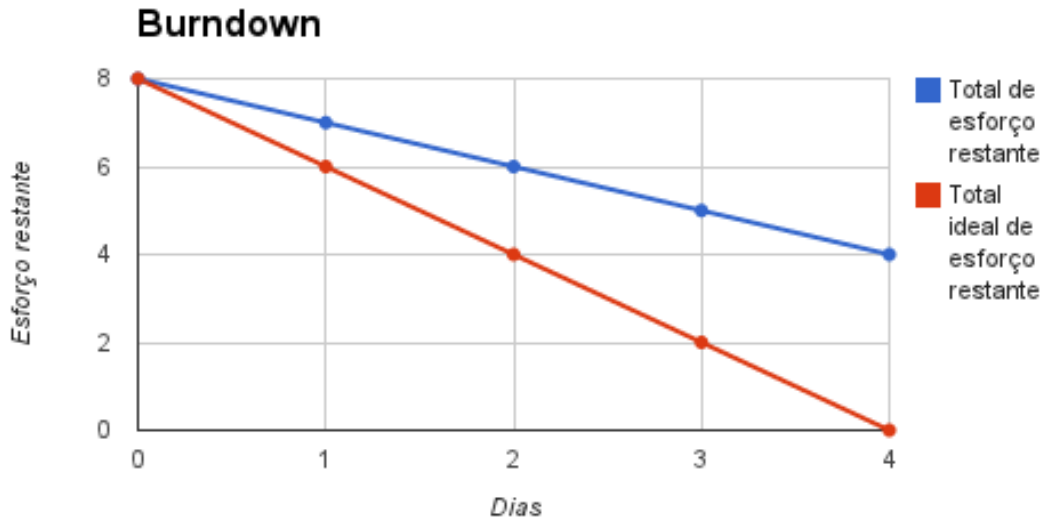

Figura 20 : Gráfico de *Burndown* para o segundo Sprint

No final do segundo Sprint foi terminado o primeiro item do *Product Backlog*, ficando o segundo por terminar devido a impedimentos ao nível da capacidade de memória dos computadores utilizados pela equipa, e do volume dos dados (Figura 20). Os resultados do Sprint foram discutidos na reunião de revisão com o *Product Owner*, e os impedimentos encontrados foram discutidos na reunião de retrospetiva.

Para o terceiro Sprint foi selecionado o segundo e terceiro item do *Product Backlog* (Tabela 19) durante a reunião de planeamento do Sprint. Nessa mesma reunião a equipa dividiu os items em tarefas a desenvolver durante o Sprint (Tabela 22).

Tabela 22 : Sprint *Backlog* para o terceiro Sprint

		Estimativa de esforço diário realizado				
Tarefa	Esforço estimado	1	2	3	4	5
Análise exploratória dos dados	2	2	0	0	0	0
Limpeza dos dados	2	0	1	1	0	0
Identificar *clusters* de clientes	2	0	0	0	0	0
Desenvolver/Adaptar o algoritmo	4	0	0	0	0	0
Total de esforço realizado por dia pela equipa		2	1	1	0	0

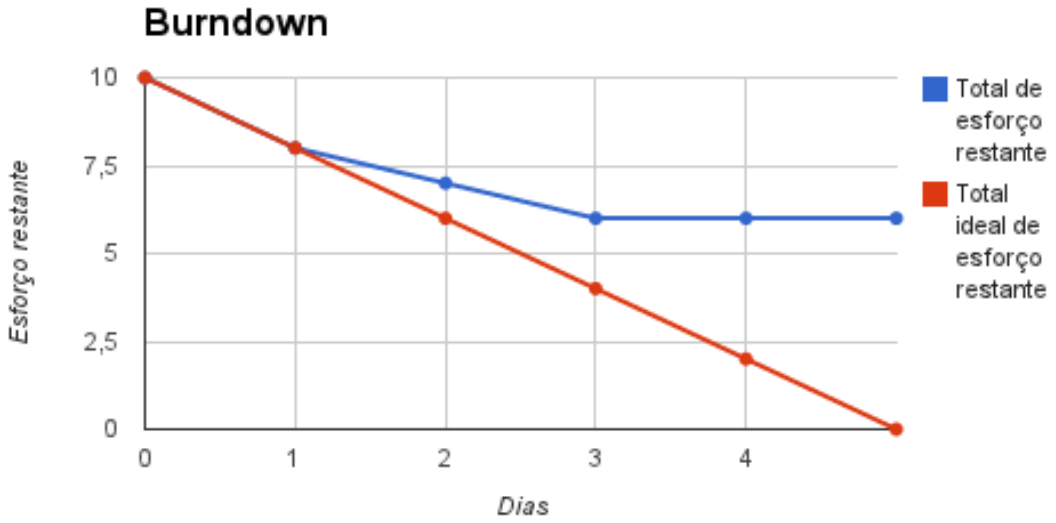

Figura 21 : Gráfico de *Burndown* para o terceiro Sprint

Durante o terceiro Sprint a equipa teve dificuldades em encontrar disponibilidade para a competição, tendo sido por essa razão apenas finalizado o segundo item do *Product Backlog* (Figura 21). No final do Sprint, a equipa teve em conjunto com o Product Owner a reunião de revisão onde foram demonstrados os resultados do Sprint e a reunião de retrospetiva.

No início do quarto Sprint foi realizada a reunião de planeamento onde a equipa selecionou o último item do *Product Backlog* para desenvolver no Sprint. Durante a reunião a equipa dividiu o item selecionado em tarefas a executar durante o Sprint (Tabela 23).

Tabela 23 : Sprint *Backlog* para o quarto Sprint

		Estimativa de esforço diário realizado				
Tarefa	Esforço estimado	1	2	3	4	5
Identificar *clusters* de clientes	4	0	0.5	0	0	1
Desenvolver/Adaptar o modelo	6	1	0.5	1	1	1
Total de esforço realizado por dia pela equipa		1	1	1	1	2

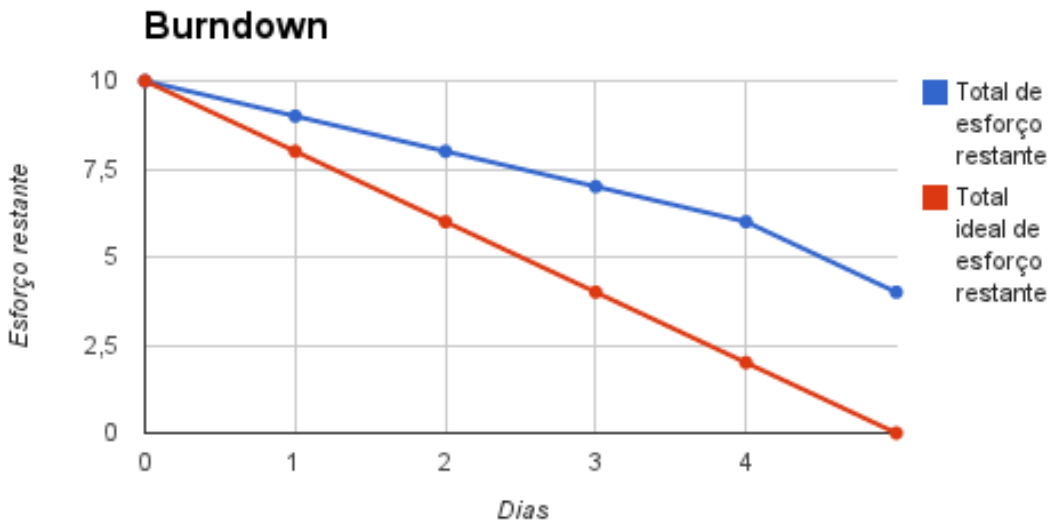

Figura 22 : Gráfico de *Burndown* para o quarto Sprint

No quarto Sprint, tal como no terceiro, a equipa não pode despender de muito tempo para a competição. Por essa razão, não foi finalizado o item selecionado para o Sprint por não ter sido atingido o critério de aceitação definido pelo *Product Owner*. No final do Sprint foram demonstrados os resultados do Sprint ao *Product Owner* na reunião de revisão e foram discutidos os impedimentos encontrados pela equipa durante o Sprint, na reunião de retrospetiva.

4.3.3 Conclusões

Com este estudo da aplicação do Scrum-DM numa competição de *data mining* verifica-se que a metodologia é aplicável em projetos nesta área, conseguindo demonstrar resultados do projeto ao cliente durante todo o processo.

Neste projeto os objetivos definidos pelo *Product Owner* não foram atingidos essencialmente por a equipa não ter tido a disponibilidade necessária para a competição. Ainda assim, a metodologia foi utilizada com sucesso pela equipa e os seus conceitos validados quanto à sua aplicabilidade.

Das entrevistas realizadas à equipa de desenvolvimento acerca da aceitação à metodologia proposta, e quais as dificuldades e impedimentos sentidos na adaptação, as conclusões foram semelhantes às verificadas no projeto descrito na Secção 4.2.

4.4 Comparação entre Scrum-DM e CRISP-DM

A metodologia proposta, Scrum-DM tem como base o CRISP-DM como metodologia para o desenvolvimento de projetos de *data mining*, e o Scrum como metodologia para a gestão ágil

do trabalho. A metodologia Scrum-DM distingue-se assim da metodologia CRISP-DM pelo facto de o desenvolvimento ser iterativo e incremental, ao contrário da metodologia CRISP-DM onde o desenvolvimento é realizado de forma sequencial, tipicamente percorrendo as seis fases da metodologia e produzindo o resultado completo de uma só vez. Na metodologia Scrum-DM uma parte do produto é desenvolvido completamente num período de duração curta, um Sprint. Isto permite que no final de cada Sprint, a equipa possa demonstrar ao cliente um resultado, e dessa forma receber feedback constante do cliente durante o desenvolvimento. No CRISP-DM há uma menor preocupação em comunicar frequentemente com o cliente final.

Outra diferença entre as metodologias, é a necessidade constante da equipa cumprir o que foi acordado no início do Sprint, respeitando os critérios de aceitação estabelecidos. Isto obriga a que o foco do desenvolvimento seja nos resultados a curto prazo, e que a detecção de erros ocorra imediatamente após acontecerem. Desta forma, reduz-se o risco para o projeto. No CRISP-DM o foco do desenvolvimento é sempre o resultado final, não estando preparado para lidar com alterações nos objetivos sem retroceder para uma fase anterior do processo, o que pode levar a um aumento do tempo necessário para surgirem resultados.

Ao contrário do CRISP-DM, o Scrum-DM permite também um acompanhamento do progresso da equipa a qualquer momento do desenvolvimento, através de uma análise do esforço restante para as tarefas planeadas para o Sprint e do gráfico de *Burndown* resultante. O Scrum-DM permite também que a qualquer momento o cliente execute as alterações que pretender ao projeto, permitindo assim um maior alinhamento dos resultados com os seus interesses.

Na Tabela 25 é apresentado um resumo das principais vantagens e os maiores desafios do Scrum-DM em relação ao CRISP-DM.

Vantagens	**Desafios**
• Aprendizagem e adaptação constante durante todo o processo; • Desenvolvimento iterativo e incremental; • Feedback do cliente em simultâneo com o desenvolvimento; • Testes contínuos pela necessidade de respeitar critérios de aceitação;	• É necessário uma equipa pequena, de dois a dez elementos; • Maior dificuldade na previsão do custo e do tempo necessário para a conclusão de um projeto; • A inexistência de um plano concreto de trabalho pode transparecer os objetivos como vagos; • Possibilidade de perder o conceito do

• Facilita interação e comunicação, entre elementos da equipa, e entre a equipa e o cliente; • Ideal para projetos de pequena dimensão onde existe a possibilidade de alterações nos objetivos durante o desenvolvimento; • Capacidade de visualização do progresso da equipa a qualquer momento do projeto.	projeto como um todo, onde o desenvolvimento iterativo pode traduzir-se em perda de qualidade final do produto. • Exige mais tempo à equipa que o CRISP-DM; • Dificuldade na definição das *Data Mining Stories* e, em especial, dos critérios de aceitação; • Dificuldade em estimar o esforço para tarefas contínuas entre Sprints.

Capítulo 5

Conclusões e Trabalho Futuro

Neste capítulo são apresentadas as conclusões obtidas através do projeto até ao momento e sugestões de trabalho futuro.

5.1 Conclusões

Partindo da necessidade dos projeto de *data mining* se tornarem mais ajustáveis a alterações nos objetivos e reduzir o tempo para a entrega de resultados ao cliente, o presente estudo tinha como objetivo o desenvolvimento de uma metodologia ágil para projetos de *data mining,* e resultados da aplicação da metodologia em projetos reais.

O desenvolvimento da metodologia foi realizado em três fases. Uma primeira fase onde foram adaptados conceitos das metodologias ágeis às metodologias para projetos de *data mining,* desenvolvendo um esboço inicial da metodologia com base da revisão da literatura. Uma segunda fase onde o esboço da metodologia foi melhorado e validado recorrendo ao método de caso retrospetivo. E finalmente uma terceira fase onde a metodologia desenvolvida foi aplicada a dois projetos de forma a confirmar a aplicabilidade da metodologia em projetos de *data mining*.

A metodologia proposta neste estudo associa à metodologia ágil Scrum para a gestão de trabalho, o CRISP-DM para o desenvolvimento de projetos de *data mining*. Possibilitando a monitorização contínua do desenvolvimento do projeto executado pela equipa, conferindo aos resultados a visibilidade necessária para que erros ou desalinhamentos com os objetivos possam ser retificados pouco depois de acontecerem. A metodologia proposta não é apenas um conjunto de práticas, mas é também um conjunto de ferramentas que permitem transparência no desenvolvimento, permitindo inspecionar os resultados no final de cada iteração e adaptá-los em relação à informação recolhida. O Scrum-DM torna visíveis disfunções e impedimentos que estão a empatar a eficácia da equipa, para que possam ser eliminados. A metodologia permite também uma maior adaptação a alterações nos objetivos, incentivando o feedback constante do cliente durante todo o processo de desenvolvimento.

Conclui-se que após a implementação da metodologia em projetos de *data mining*, foi possível um maior controlo do desenvolvimento dos projetos, uma maior aceitação à alteração de objetivos e requisitos permitindo uma melhoria no alinhamento dos projetos com os interesses do negócio, e uma maior flexibilidade para a equipa desenvolver como pretende e à sua velocidade, respeitando os objetivos.

5.2 Trabalho Futuro

Durante a realização deste estudo surgiram algumas ideias que poderiam ser exploradas, mas que por limitação de tempo não foram desenvolvidas. São aconselhados mais estudos da aplicação da metodologia proposta, em projetos com uma equipa de dimensão superior a dois elementos de forma a verificar como se adaptam equipas de grande dimensão à metodologia. É também aconselhado o estudo da utilização da metodologia com a mesma equipa em projetos consecutivos de forma a verificar se a capacidade de trabalho por Sprint de um projeto pode ser extrapolado para um posterior.

Aconselha-se também o desenvolvimento de um sistema de informação para apoio ao Scrum-DM e de um guia rápido à metodologia, com duas perspetivas, a de quem tem conhecimento da aplicação comum do Scrum na engenharia de software e pretende perceber de que forma pode aplicar o Scrum a um projeto de *data mining*, e a perspetiva de quem entende o processo de desenvolvimento de um projeto de *data mining* e pretende entender como agilizar o processo de desenvolvimento e gestão do trabalho em projetos de *data mining*.

Referências

Agile Software Development, CSA Case Study , Saad Ashmawi, 2013

Yu Beng Leau, Wooi Khong Loo, Wai Yip Tham and Soo Fun Tan. 2012. "Software Development Life Cycle AGILE vs Traditional Approaches". International Conference on Information and Network Technology (ICINT 2012)

Everette R. Keith. December 2002. Agile Software Development Processes A Different Approach to Software Design

Mouhib Alnoukari , Zaidoun Alzoabi , Saiid Hanna. Applying Adaptive Software Development (ASD) Agile Modeling on Predictive *Data mining* Applications: ASD-DM Methodology. Information Technology, 2008. ITSim 2008. International Symposium on (Volume 2). doi: 10.1109/ITSIM.2008.4631695

Farbey, B; Finkelstein, A; 2001 Evaluation in software engineering: ROI, but more than ROI.

A B M Moniruzzaman , Dr. Syed Akhter Hossain. Comparative Study on Agile Software Development Methodologies. Global Journal of Computer Science and Technology Volume XIII Issue VII Version I Year 2013

Agile Manifesto. 2001 "Manifesto for Agile Software Development". Acedido a 10 de Janeiro de 2014. http://agilemanifesto.org/

An Introduction to AgIle SoftwAre development. Junho 2007. Acedido a 11 de Janeiro de 2014. http://www.serena.com

Ana Azevedo, Manuel Filipe Santos. 2008 IADIS. "KDD, SEMMA and CRISP-DM: A Parallel Overview". ISBN: 978-972-8924-63-8.

Rayid Ghani , Carlos Soares. KDD-2006 Workshop. "*Data mining* for Business Applications".

Referências

Chris Voss, Nikos Tsikriktsis and Mark Frohlich. 2002. “Case research in operations management”. London Business School, London, UK

Natalia Juristo, Ana M. Moreno. 2010. Basics of Software Engineering Experimentation. Universidad Politéncnica de Madrid Spain. ISBN:1441950117 9781441950116

Ahlstrom Karlsson. 2008. “Researching Operations Management”, 60-83

Ahlstrom Karlsson. 2009. “Longitudinal Field Studies”, 196-235

Pete Chapman (NCR), Julian Clinton (SPSS), Randy Kerber (NCR), Thomas Khabaza (SPSS), Thomas Reinartz (DaimlerChrysler), Colin Shearer (SPSS) and Rüdiger Wirth (DaimlerChrysler). 2010. “CRISP-DM 1.0 Step-by-step data mining guide”

Rüdiger Wirth , Jochen Hipp. 2000. “CRISP-DM: Towards a Standard Process Model for *Data mining*”

Jiawei Han, Micheline Kamber. 2006. “Data mining: Concepts and Techniques”. Second Edition, University of Illinois at Urbana-Champaign

Ron Zacharski. 2013. “Programmer’s Guide to *Data mining*: The Ancient Art of the Numerati”. Acedido a 4 de Fevereiro de 2014. www.guidetodatamining.com

Yongjian Fu. 2008. *Data mining*: Tasks, ,Techniques, and Application. Department of Computer Science, University of Missouri - Rolla

Colin Shearer. 2000. “The CRISP-DM Model: The New Blueprint for DataMining”. JOURNAL of Data Warehousing, Volume 5, Number 4, 13-22

SAS Enterprise. 2008 “SEMMA”. Acedido a 10 de Dezembro de 2013.
http://www.sas.com/offices/europe/uk/technologies/analytics/datamining/miner/semma.html

Wendorff, Peter. 2002. “An Essential Distinction of Agile Software Development Processes Based on Systems Thinking in Software Engineering Management.”.

Cássio O. Camilo, João C. Silva. 2009. “Mineração de Dados: Conceitos, Tarefas, Métodos e Ferramentas“. Instituto de Informática Universidade Federal de Goiás

Referências

Kent Beck, 1999. “Extreme Programming Explained: embrace change”. First Edition September 29. Addison-Wesley.

Mingers J. , Borcklesby, J. 1997. “Multimethodology: towards a framework for mixing methodologies.”. Omega: International Journal of Management Science, 289-507

Gill , Johnson. 1991. “Research Methods for Managers. London. Paul Chapman Publishing”

Eisenhardt, K. 1989. “Building Theories From Case Research”. Academy of Management Review, 532-550

Emery J. 1973. “Cost Benefit Analysis of Information Systems”, SMIS Workshop, no.1

Parker, M.M., R.J. Benson and H.E. Trainor, Information Economics: Linking business performance to information technology, Englewood Cliffs., 1998 Prentice Hall

Rivard E. and Kaiser K. The benefits of quality IS, Datamation, (January, 1989), 53-58

Kenny, R.L. and Raiffa, H. Decisions with multiple objectives: preferences and value tradeoffs, (New York, 1976) John Wiley and Sons

Diane E. Strode. 2005. “The Agile Methods: An Analytical Comparison of Five Agile Methods and an Investigation of Their Target Environment”. Master in Information Sciences. Massey University, Palmerston North, New Zealand. http://mro.massey.ac.nz/bitstream/handle/10179/515/02whole.pdf

D. L. Olson, D. Delen, 2008. “Schematic of SEMMA”. Advanced *Data mining* Techniques, 19. Softcover

Cockburn, A. 2004. “Crystal’s coverage of diferente project types”. Crystal Clear: A Human-Powered Methodology For Small Teams, including The Seven Properties of Effective Software Projects, 240

Cohn, M.W. Selecting an Agile Process: Choosing Among the Leading Alternatives. Proceeding of SD Best pratices; conference & expo 2004, September 21, 2004

Yin, R.K. (2003), Case Study Research: Design and Methods, Applied Social Research Methods Series, Volume 5, Sage Publications, 3rd ed., Thousand Oaks

Referências

F. Pinto, C. Soares, Space Allocation in the Reatil Industry: A Decision Support System Integrating Evolutionary Algorithms and Regression Models. 2012

Hawkins, D. Identification of outliers. Monographs on Applied Probability and Statistics. 1980

E. Gummesson, Qualitative Methods in Management Research, Sage, 2000.

Jossey Bass. The Leader's Guide to Radical Management: Reinventing the Workplace For the 21st Century. 2010.

Anexo A

Evidência recolhida por observação

7.1 Data Intelligence Group

Nº	Data	Resumo da reunião	Problemas identificados	Possíveis soluções
1	19/02/2014	Reunião para iniciar o projeto. Nesta reunião foi descrito o problema a resolver. O objetivo do projeto é prever avarias nos componentes dos computadores da marca ASUS (previsão para 1.5 anos). Foram também definidas as tarefas a realizar até à próxima reunião: • Análise exploratória dos dados • Geração dos dados completos (faltam só casos com 0 componentes com avarias)	Dificuldade na definição do problema. Tentativa de alguns elementos em realizar tarefas posteriores à análise exploratória, antes de realizar a mesma.	Utilização de *user stories* para definição do problema e objetivos para o *data mining*.

Evidência recolhida por observação

2	26/02/2014	Nesta reunião foi demonstrada a análise exploratória dos dados, realizada até ao momento. Análise gráfica. Houve também uma tentativa de estabelecer uma *baseline*. Para a próxima reunião a tarefa é a preparação dos dados, nomeadamente agregação de dados e início do desenvolvimento dos modelos. Dividiu-se também o DIG em 3 equipas e foi definida a forma de validação dos modelos a nível interno(utilizar o ano de 2009).	Dificuldade na definição de tarefas e na cooperação e partilha de dados entre a equipa.	Utilização de Sprint *backlog* com tarefas bem definidas para o Sprint.
3	05/03/2014	Análise dos modelos desenvolvidos até ao momento e identificação das possíveis tarefas a desenvolver para os melhorar.	Dificuldade em definir e alocar tarefas para os elementos da equipa.	Utilização de Sprint *backlog* com tarefas bem definidas para o Sprint.
4	12/03/2014	Análise dos modelos desenvolvidos até ao momento, pela equipa, e identificação das possíveis tarefas a desenvolver para os melhorar.	Dificuldade em definir e alocar tarefas para os elementos da equipa.	

7.2 Projeto de *Data Mining*

Nº	Data	Resumo da reunião	Problemas identificados	Possíveis soluções
1	25/02/2014	Reunião para iniciar o projeto. Nesta reunião foi descrito o	Dificuldade na definição dos	Utilização de *user stories*

Evidência recolhida por observação

		problema a resolver na perspetiva do negócio e foi realizada uma tentativa de definição dos objetivos para o *data mining*. Foi também realizado o planeamento macro para todo o projeto (definição das principais fases e *milestones*).	objetivos para o *data mining*.	de forma a facilitar a definição dos objetivos para o *data mining* por parte do negócio.

7.3 Projeto de *Business Intelligence*

Nº	Data	Resumo da reunião
1	28/02/2014	Reunião de arranque do projeto. Para a gestão do projeto será utilizados bastantes conceitos do scrum, nomeadamente: *product owner, scrum master, weekly scrum*, reunião mensal de planeamento do Sprint com Sprint *review* e *retrospective*. A reunião de scrum é semanal.
2	07/03/2014	Reunião de scrum semanal: Foram identificados erros no planeamento inicial para o esforço das tarefas do Sprint (esforço e tempo tem de ser inferior a uma semana). Foram verificadas as tarefas que foram concluídas, as tarefas planeadas e impedimentos que tenham surgido ou possam surgir na execução das mesmas.
3	14/03/2014	Reunião de scrum semanal: Foi verificado se as tarefas planeadas correram sem problemas, se estão concluídas, ou caso contrário, que problemas surgiram e de que forma os podem resolver.
4	21/03/2014	Reunião de scrum semanal: Verificação se as tarefas propostas para a semana foram cumpridas. Problemas encontrados e de que forma os solucionar.
5	28/03/2014	Reunião de scrum semanal: Revisão do que foi realizado em relação ao proposto. Verificação dos bloqueios que existam e de que forma os solucionar.
6	01/04/2014	Reunião de planeamento do Sprint:

		Definição das tarefas para o segundo Sprint. Atribuição das tarefas aos elementos da equipa e definição da carga horária para cada elemento por tarefa.
7	11/04/2014	Reunião de scrum semanal: Foi verificado se as tarefas planeadas correram sem problemas, se estão concluídas, ou caso contrário, que problemas surgiram e de que forma os podem resolver.
8	23/04/2014	Foi realizada uma demonstração interna do trabalho realizado até ao momento (os dois protótipos funcionais, um para Microsoft Server, e outro para *Pentaho*); Esta demonstração foi realizada para avaliação do *product owner*
9	02/05/2014	Foi finalizada a apresentação dos protótipos funcionais.
10	09/05/2014	Reunião de Scrum Semanal: Verificação das tarefas planeadas para a semana anterior, tarefas finalizadas e tarefas por finalizar. Das tarefas por finalizar foi verificado quais os bloqueios que impediram o desenvolvimento.
11	16/05/2014	Reunião de Scrum Semanal: Revisão das tarefas previstas para estarem concluídas nesta semana; Das que não estão concluídas, foram verificados os impedimentos e como os ultrapassar.
12	23/05/2014	Reunião de Scrum Semanal: Revisão das tarefas que tinham sido previstas para a semana; Planeamento das entregas ao cliente, workshops e o manual de utilização.
13	13/06/2014	Nesta reunião foi definido um novo Sprint, planeadas as tarefas a serem realizadas para a próxima semana e estimado o esforço por tarefa para cada elemento da equipa.

Anexo B

Guião para as entrevistas dos estudos de caso retrospetivos

Eu sou o Diogo Nogueira, aluno do Mestrado Integrado em Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto. A presente investigação decorre no âmbito do Projeto de Dissertação para a obtenção do grau de Mestre.

O meu tema do Projeto de Dissertação é Agile Data Mining: Uma metodologia ágil para o desenvolvimento de projetos de *data mining*.

Este estudo visa explorar projetos de *data mining*, dos entrevistados, que já tenham sido finalizados e a metodologia utilizada para o desenvolver, de forma a realizar uma aplicação retrospetiva da nova metodologia desenvolvida.

A entrevista tem uma duração estimada de 15minutos.

- Nome do entrevistado e responsabilidade
- Nome do projeto
- Hora e local

Questões:

1. Projeto

1.1. Descreva por favor os objetivos do projeto.

1.1.1. Quais os objetivos do negócio?

1.1.2. Qual a motivação/justificação?

1.1.3. Quais os objetivos para o *data mining*?

1.2. Como foram os objetivos definidos inicialmente?

1.2.1. Foram definidos pelo negócio? Ou existiu colaboração com a equipa de desenvolvimento?

1.2.2. Caso tenha existido colaboração, de que forma foi realizada?

1.2.3. Existiu dificuldade na definição dos objetivos?

2. Metodologia

2.1. Qual foi a metodologia aplicada no desenvolvimento?

2.1.1. CRISP-DM ou SEMMA?

2.1.2. Caso nenhuma das anteriores, pode descrever o processo aplicado e as fases envolvidas?

2.3. No fim de cada fase existia claramente um resultado?

2.2. Foram definidas tarefas de forma colaborativa entre os elementos da equipa?

2.2.1. Caso contrário, de que forma foram definidas?

2.3. A equipa envolvida no projeto era de quantos elementos?

2.3.1 Caso a equipa seja constituída por mais do que um elemento:

2.3.2. Existiram tarefas em paralelo atribuídas a diferentes elementos da equipa?

2.3.3. Como foi realizada a distribuição das tarefas pelos elementos da equipa?

2.3. Qual foi a duração do projeto?

2.4. Qual foi a duração de cada fase?

2.5. Como foi coordenado o projeto?

2.5.1. Existiram reuniões de planeamento e análise do que foi realizado de forma cíclica?

2.5.2. Nessas reuniões (cíclicas ou não) havia presença de alguém que representasse o negócio?

2.5.3. Quem estava envolvido nessas reuniões?

2.6. Em que fases do projeto existiu documentação?

2.7. Depois da fase inicial de definição dos objetivos e de planeamento (*business understanding*), existiram ciclos de desenvolvimento? Por exemplo, depois do desenvolvimento de um modelo e análise dos resultados houve necessidade de regressar para uma fase anterior do processo?

2.7.1. Caso tenham existido ciclos, qual foi aproximadamente a sua duração média?

2.8. Foi realizado o *deployment* (integração) dos resultados do *data mining*?

2.8.1. Em caso afirmativo, no que consistiu o *deployment*?

2.8.2. De que forma foi realizado o seu planeamento?

2.8.3. Quais as tarefas envolvidas?

2.9. Durante o projeto, existiram alterações nos objetivos para o negócio e/ou dos objetivos para o *data mining*?

2.9.1. No caso de terem existido alterações, as tarefas foram replaneadas de forma colaborativa entre a equipa e o negócio?

2.10. Durante o projeto surgiram novas questões de negócio que originaram um novo processo de desenvolvimento ou que poderiam ter originado?

Anexo C

Guião para entrevistas dos estudos de caso da aplicação da metodologia Scrum-DM

Eu sou o Diogo Nogueira, aluno do Mestrado Integrado em Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto. A presente investigação decorre no âmbito do Projeto de Dissertação para a obtenção do grau de Mestre.

O meu tema do Projeto de Dissertação é Agile Data Mining: Uma metodologia ágil para o desenvolvimento de projetos de *data mining*.

Este estudo visa explorar o grau de aceitação e as principais dificuldades e impedimentos, dos entrevistados, da adaptação à metodologia ágil proposta para projetos de *data mining*.

A entrevista tem uma duração estimada de 30 minutos.

- Nome do entrevistado
- Nome do projeto
- Hora e local

Questões:

1. Qual a metodologia de desenvolvimento de projetos de *data mining* que utilizava antes de aplicar a metodologia ágil proposta?
2. Dos conceitos da metodologia ágil, quais os que achou mais difíceis de definir e porquê?
   2.1. *Data Mining Stories*
   2.2. Critérios de aceitação
   2.3. Priorização do *Product Backlog*
   2.4. Definição do Sprint *Backlog*
   2.5. Estimação do esforço dos itens do *Product Backlog*

2.6. Estimação do esforço das tarefas do Sprint *Backlog*

3. Quais as vantagens e desvantagens que encontra com a utilização da metodologia ágil em relação à que utilizava anteriormente?

4. Considera que a equipa se adequou ao desenvolvimento seguindo a metodologia?

5. Quais as vantagens e desvantagens que encontra com os conceitos da metodologia ágil?

5.1. *Data Mining Stories*

5.2. Critérios de aceitação

5.3. *Product Backlog*

5.4. Sprint *Backlog*

5.5. Estimação do esforço

5.6. Gráfico de *Burndown*

6.No final de cada Sprint considera que é possível na área do *data mining*, ter sempre um resultado que possa ser demonstrado ao *Product Owner?*

6.1. Em caso negativo, porquê?

7. Como classifica a facilidade de definição das *Data Mining Stories* e seus critérios de aceitação?

7.1. Considera que são úteis?

8. Na sua opinião qual foi o foco de desenvolvimento durante a utilização diária da metodologia ágil?

9.Comentários adicionais acerca da metodologia.

Anexo D

Evolução do *Product Backlog* no projeto de *data mining*

Product Backlog atualizado no início do segundo Sprint

Prioridade	*Data mining Story*	Critério de aceitação	Estimativas iniciais de esforço
1	Como analista, quero criar variáveis que permitam definir o perfil de clientes e avaliar os *clusters* resultantes	Conjunto de variáveis para definição de perfis em tempo útil, com comparação de desempenho	40 horas
2	Como analista, quero uma ferramenta de visualização de chamadas de forma a aumentar o conhecimento da base de dados dos registos.	Protótipo da interface de visualização de chamadas	40 horas
3	Como analista, quero ter disponíveis todos os resultados dos *clusterings* já realizados de forma a perceber a sua evolução	Cinco *clusters* guardados na base de dados	40 horas
4	Como analista, quero definir um conjunto de estatísticas robustas de avaliação de	Relatório com a definição e evolução de resultados das diferentes	40 horas

	chamadas, de forma a caracterizar os registos anómalos.	estatísticas ao longo de um mês de dados.	
5	Como analista, quero ter um método de detecção de anomalias com base num algoritmo de *clustering* incremental de forma a detetar chamadas anómalas	Resultados da detecção de anomalias para um dia de chamadas com pelo menos uma chamada anómala detetada	30 horas
6	Como analista, quero definir, modelar e comparar diferentes mecanismos de avaliação de falhas de forma a incorporar os registos com os algoritmos de *clustering*.	Relatório com os diferentes resultados, tendo em conta um mês de registos.	30 horas

Product Backlog atualizado no início do terceiro Sprint

Product Backlog depois da reunião de planeamento do terceiro Sprint, priorizado pelo *Product Owner*.

Prioridade	*Data mining Story*	Critério de aceitação	Estimativas iniciais de esforço
1	Como analista, quero ter disponíveis todos os resultados dos *clusterings* já realizados de forma a perceber a sua evolução	Cinco *clusters* guardados na base de dados	28 horas
2	Como analista, quero uma ferramenta de visualização de chamadas de forma a aumentar o conhecimento da base de dados dos registos.	Protótipo da interface de visualização de chamadas	40 horas
3	Como analista, quero criar variáveis que permitam definir o perfil de clientes e avaliar os *clusters* resultantes	Conjunto de variáveis para definição de perfis em tempo útil, com comparação de desempenho	12 horas
4	Como analista, quero definir um conjunto de estatísticas robustas de avaliação de	Relatório com a definição e evolução de resultados das diferentes	40 horas

	chamadas, de forma a caracterizar os registos anómalos.	estatísticas ao longo de um mês de dados.	
5	Como analista, quero ter um método de detecção de anomalias com base num algoritmo de *clustering* incremental de forma a detetar chamadas anómalas	Resultados da detecção de anomalias para um dia de chamadas com pelo menos uma chamada anómala detetada	30 horas
6	Como analista, quero definir, modelar e comparar diferentes mecanismos de avaliação de falhas de forma a incorporar os registos com os algoritmos de *clustering*.	Relatório com os diferentes resultados, tendo em conta um mês de registos.	30 horas

Product Backlog atualizado no início do quarto Sprint

Prioridade	*Data mining Story*	Critério de aceitação	Estimativas iniciais de esforço
1	Como analista, quero definir, modelar e comparar diferentes mecanismos de avaliação de falhas de forma a incorporar os registos com os algoritmos de *clustering*.	Relatório com os diferentes resultados, tendo em conta um mês de registos.	6 horas
2	Como analista, quero criar variáveis que permitam definir o perfil de clientes e avaliar os *clusters* resultantes	Conjunto de variáveis para definição de perfis em tempo útil, com comparação de desempenho	6 horas
3	Como analista, quero ter um método de detecção de anomalias com base num algoritmo de *clustering* incremental de forma a detetar chamadas anómalas	Resultados da detecção de anomalias para um dia de chamadas com pelo menos uma chamada anómala detetada	18 horas
4	Como analista, quero definir um conjunto de estatísticas robustas de avaliação de chamadas, de forma a caracterizar	Relatório com a definição e evolução de resultados das diferentes estatísticas ao longo de um	18 horas

	os registos anómalos.	mês de dados.	
5	Como analista, quero um método para visualização de dois resultados de *clustering* de forma a realizar comparações	Resultados da comparação entre *clusters*	30 horas

Product Backlog atualizado no início do quinto Sprint

Product Backlog depois da reunião de planeamento do quinto Sprint.

Prioridade	*Data mining Story*	Critério de aceitação	Estimativas iniciais de esforço
1	Como analista, quero definir um conjunto de estatísticas robustas de avaliação de chamadas, de forma a caracterizar os registos anómalos.	Relatório com a definição e evolução de resultados das diferentes estatísticas ao longo de um mês de dados.	30 horas
2	Como analista, quero um método para visualização de dois resultados de *clustering* de forma a realizar comparações	Resultados da comparação entre *clusters*	30 horas