

Ana Teresa Moniz Pimentel

Impacto do Lean IT na execução de projetos:

Caso de estudo no setor bancário

Orientador: Professor Doutor Paulo Jorge Tavares Guedes

Universidade Lusófona de Humanidades e Tecnologias

Escola de Comunicação, Artes e Tecnologias da Informação (ECATI)

Departamento de Engenharia Informática e Sistemas de Informação

Lisboa

2018

Ana Teresa Moniz Pimentel

Impacto do Lean IT na execução de projetos:

Caso de estudo no setor bancário

Dissertação defendida em prova pública na Universidade Lusófona de Humanidades e Tecnologias, no dia 22 de Maio, perante o júri nomeado pelo Despacho de Nomeação nº 156/2018, de 11 de Abril, com a seguinte composição:

Presidente: Professor Doutor José Luís Azevedo Quintino Rogado (ULHT)

Arguente: Professora Doutora Maria do Rosário Gomes Osório Bernardo Ponces de Carvalho (IST/UL)

Orientador: Professor Doutor Paulo Jorge Tavares Guedes (ULHT)

Universidade Lusófona de Humanidades e Tecnologias
Escola de Comunicação, Artes e Tecnologias da Informação (ECATI)
Departamento de Engenharia Informática e Sistemas de Informação

Lisboa

2018

“A simplicidade é o último degrau da sabedoria.”

KHALIL GIBRAN

Agradecimentos

Ao Doutor Paulo Guedes

Ao João Cruz

Aos meus pais

Obrigada por todo o apoio.

Resumo

O planeamento rigoroso, o controlo e a gestão de mudança há muito que deixaram de ser garantia suficiente para vingar no mundo das Tecnologias de Informação. A vantagem competitiva passa por ter uma boa capacidade de inovação conseguida através da redução do desperdício e vai permitir reposicionar os recursos para uma rápida resposta à mudança.

A abordagem Lean no IT abre a possibilidade de identificação de um diversificado conjunto de oportunidades de melhoria de gestão que poderão traduzir-se na diminuição do desperdício, no aumento da eficiência dos processos e ganhos de satisfação dos clientes. Este trabalho incidirá na análise de dois projetos e a metodologia adotada pretende avaliar o impacto da abordagem Lean IT opondo-a à gestão convencional, comparando vantagens e desvantagens e ilustrando com os principais KPIs dos respetivos projetos.

Das principais conclusões deste estudo destaca-se que a aplicação do Lean IT se traduziu num aumento de produtividade estimado em 23% em apenas dois anos e meio de aplicação ao projeto, permitindo, simultaneamente, aumentar o volume das tarefas executadas pelas equipas e melhorar a eficiência na sua realização. Em síntese, a adoção do Lean IT ganhou substanciais em termos de custos de time-to-market.

Palavras-chave: Lean IT, Gestão de Sistemas de Informação; Gestão de Projeto

Abstract

Rigorous planning, control, and change management have long ceased to be sufficient assurances to avenge in Information Technology world. The competitive advantage is to have a good innovation capacity achieved through the reduction of waste and will allow repositioning the resources for a rapid response to change.

The Lean approach in IT opens up the possibility of identifying a diverse set of management improvement opportunities that can translate into reduced waste, increased process efficiency, and customer satisfaction gains. This work will focus on the analysis of two projects and the methodology adopted intends to evaluate the impact of the Lean IT approach, opposing it to conventional management, comparing advantages and disadvantages and illustrating with main KPIs of each projects.

The main conclusions of this study are that the application of Lean IT resulted in an increase in productivity estimated at 23% in only two and a half years of application to the project, while simultaneously increasing the volume of tasks performed by the teams and improving the efficiency. In summary, the adoption of Lean IT translates into significant gains in time-to-market.

Keywords: Lean IT, Information Systems Management; Project management

Abreviaturas, siglas e símbolos

AIF – Aplicativo das Instituições Financeiras

ALM - Application Lifecycle Management

ASD - Adaptive Software Development

BI – Business Intelligence

CCTA - Central Computer and Telecommunications Agency

CMM - Capability Maturity Model

CMMI - Capability Maturity Model Integration

COBIT - Control Objectives for Information and Related Technologies

DRO – Departamento de regulação e Organização do Sistema Financeiro

DSDM - Dynamic systems development method

DSI – Departamento de Supervisão Instituições Financeiras

DSM – Design Structure Matrix

DTI – Departamento Tecnologias de Informação

EDT – Estrutura de decomposição de trabalho

ESI – Espírito Santo Informática

ETL – Extract Transform and Load

EUA – Estados Unidos da América

FDD - Feature Driven Development

FIFO – First in first out

IAS - International Accounting Standard

IF – Instituição Financeira

IFRS - International Financial Reporting Standards

IT – information technology

ITIL - Information Technology Infrastructure Library

KPI - Key Performance Indicator

LDAP – Lightweight Directory Access Protocol

MoSCoW - Must have, Should have, Could have, and Won't have

NATO - North Atlantic Treaty Organization

NBSI – Novo Banco Sistemas de Informação

OID – Oracle Internet Directory

OOPSLA - Object-Oriented Programming, Systems, Languages and Applications

PMBOK - Project Management Body of Knowledge

PMI - Project Management Institute

SCM – Source Code Management

SDLC - System Development Lifecycle

SEPG - Software Engineering Process Group

SI – Sistemas de Informação

SLA – Service level Agreement

SOA - Service-oriented architecture

SPC – Single Point of Contact´

SPT – Sistema de Produção da Toyota

SVN - Subversion

VBA – Visual Basic For Applications

VCB – Visual Control Boards

VSM – Value Stream Map

XP – eXtreme Programming

Índice

Agradecimentos	4
Resumo	5
Abstract	6
Abreviaturas, siglas e símbolos	7
Índice de Figuras	10
Introdução	11
1 Estado da Arte	15
1.1 Modelo em Cascata	15
1.2 Modelos de Maturidade IT: CMMI e ITIL.....	24
1.3 Agile	27
1.4 Comparação Gestão Tradicional e Agile.....	42
1.5 Lean.....	46
2 Abordagem de Investigação	69
3 Caso de estudo	71
3.1 Análise do caso A – Gestão tradicional de projetos	71
3.2 Análise do caso B – Gestão Lean IT de projetos.....	104
4 Conclusão	125
5 Bibliografia	130

Índice de Figuras

Figura 1: Desenho esquemático do modelo de desenvolvimento em cascata (Sommerville, 2011)	17
Figura 2: Comparação ITIL e CMMI (Adaptado de SEPG, 2008).....	26
Figura 3: Ciclo de vida de um projeto Agile - Adaptado Hass (2007)	30
Figura 4: Adotação do triângulo de ferro tradicional, publicado originalmente por Martin Barnes em 1969 (PMI, 2013).....	44
Figura 5: Estrutura organizacional dos envolvidos no projeto AIF	78
Figura 6: Matriz de Avaliação de Riscos (Adaptado de Miguel, 2015)	82
Figura 7: Cronograma do projeto AIF (estimado e real)	84
Figura 8: Detalhe do documento com a Estrutura de Decomposição de Trabalho do projeto em estudo	87
Figura 9: Diagrama do fluxo que descreve o fluxo de informação entre as várias camadas e aplicações.....	93
Figura 10: Fonte (ESI/NBI)	105
Figura 11: Cronologia da Transformação Lean IT (Fonte: ESI/NBSI)	107
Figura 12: Processo de implementação do Lean IT numa equipa	107
Figura 13: Alocação dos recursos ESI antes da abordagem Lean. Adaptado de Aula GSTI Cap 3.3 Lean	109
Figura 14: Organização da equipa por pools em consequência da transformação Lean IT. Adaptado de GSTI Cap 3.3 Lean	110
Figura 15: Dois tipos de gráficos muito utilizados para seguir e comunicar o progresso dos projetos, os gráficos de Burndown e de Burn-Up.....	113
Figura 16: Whiteboard utilizado pelas equipas Lean IT do ESI/NBSI.....	114

Introdução

Com mais de cinco décadas de experiência e desenvolvimento a engenharia de *software* apresenta-se hoje, mais do que nunca, como um desafio permanente e empolgante para equipas e gestores. Com o correr do tempo, a produção de novos e cada vez mais diversificados sistemas de informação passou a ter um conjunto alargado de clientes de quase todos os setores da economia e da sociedade, tornando-se uma atividade inerentemente complexa, com várias dependências, que requer *inputs* de diversos indivíduos de áreas de conhecimento distintas e em geral não sobreponíveis. A autêntica revolução por que passou e passa, obriga a engenharia do *software* a um esforço de coordenação e gestão marcado por uma dinâmica imposta pelo próprio mercado (Maruping, Venkatesh & Agarwal, 2009).

Atormentada por taxas de sucesso baixas ao longo de décadas consecutivas, a indústria de *software* é das indústrias que apresentam valores de produtividade mais baixos (Hibbs, Jewett & Sullivan, 2009). Segundo Standish Group, publicado no CHAOS Report de 2016, apenas 9% dos projetos de *software* considerados de tamanho médio foram concluídos com sucesso e quando se olha para o número mas agora de grandes projetos a taxa de sucesso decai para 2% (Standish Group, 2016).

Para enfrentar os constantes obstáculos e desafios a que estão sujeitas, nomeadamente em áreas como a da qualidade do *software* disponibilizado e da eficiência no desempenho, as empresas têm vindo a desenvolver diferentes metodologias de gestão do processo de desenvolvimento, tornando-se algumas delas, como ITIL e CMMI, standards corporativos (Ziolhowski & Deregowski, 2014).

Paralelamente, as produtoras de *software* procuram continuamente e elaboram programas de melhoria de processos destinados a otimizar os métodos subjacentes e a se tornarem cada vez mais eficientes recorrendo muitas das vezes a esses standards corporativos (Bass, 2003).

Convencionalmente as metodologias que defendem planeamento rigoroso e pressupõem previsibilidade nas atividades (Boehm, 2002) revelam dificuldades de resposta à rápida mudança de requisitos fortemente impulsionada pelas necessidades de negócio em contínua evolução (Maruping et al., 2009) e ao mesmo tempo vivem a

tentar sobreviver no limiar do caos (Schmidt & Lyle 2010). Como os sistemas de informação pertencem a um mundo complexo e por vezes frágil, onde uma pequena falha pode destabilizar o sistema total, os tradicionalistas defendem a ideia que o IT deve manter o seu papel de estabilizador (Bell, 2006) apesar da gestão da mudança ser um processo lento onde tudo é meticulosamente colocado em produção para prevenir ao máximo o tempo de inatividade e disrupção do negócio (Hibbs et al, 2009).

Segundo Bell (2006), a principal razão dos insucessos do IT nos dias de hoje não se deve diretamente ao facto de ser incapaz de acompanhar a mudança, este autor defende que o sucesso seria maior se houvesse um maior alinhamento do IT com o negócio. A crescente complexidade do mercado e das tecnologias vieram complicar os processos de negócio que tentam mimetizar a complexidade exterior, acumulando ao mesmo tempo, e por receio de disrupção, comportamentos antigos com os atuais. Estes processos foram então criados desnecessariamente complexos e os sistemas de informação que os tentam suportar, são inevitavelmente e desnecessariamente complexos. Toda esta complexidade não traz necessariamente mais benefícios que custos (Bell e Orzen, 2010).

Como principal contraponto aos métodos tradicionais surgiu a abordagem Agile, que no início do ano de 2009 derivado à crise económica foi adotada por inúmeras empresas em todo o mundo de modo a se tornarem mais eficientes no desenvolvimento do *software* (Elbert et al., 2012). Apesar dos esforços do Agile terem trazido melhorias tremendas com a nova abordagem focada no cliente, a sua aplicabilidade à empresa como um todo permaneceu estagnada (Elbert et al., 2012). Em resposta a esta lacuna surgiu o Lean IT, cujas práticas quando aplicadas de forma disciplinada e sistemática permitem orquestrar o alinhamento do IT com o negócio de forma sustentável (Bell e Orzen, 2010)

Será na indústria automóvel que surgirão os pioneiros do Lean e o seu aparecimento surge associado ao reconhecimento de que o desperdício gerado pelo fabrico de um produto deve ser rapidamente identificado e eliminado, uma vez que se não adiciona qualquer valor para o cliente, inibe a qualidade e reduz a rentabilidade (Waterhouse, 2008). O Lean tem um histórico comprovado de melhoria simultânea do custo, qualidade, velocidade e agilidade de produção. (Gartner, 2008). Segundo Staats, Brunner & Upton, (2011), o sistema de produção Lean pode também ser implementado no trabalho qualificado.

Num contexto em que as tecnologias de informação, os seus processos, o desenvolvimento de *software* e as suas atividades *umbrela* necessitam de manter um ritmo acelerado para alavancar a produtividade num mundo em que os negócios apresentam uma natureza multifacetada e os sistemas de IT que os suportam se tornaram indispensáveis, cada vez mais complexos (Hass, 2007) e que carecem de simplificação (Bell & Orzen, 2010) ganha relevo a análise das metodologias de produção de sistemas de informação que crescentemente colonizam o dia-a-dia do cidadão comum.

A presente dissertação visa, em particular, a avaliação do impacto da abordagem Lean IT opondo-se à gestão convencional de projetos, recorrendo à metodologia de estudo de caso que pretende ilustrar duas realidades, comparando vantagens e desvantagens de ambas e complementando com alguns KPIs de projeto de modo a caracterizar o seu desempenho. Muitos autores defendem que ainda existem poucos casos de estudo empíricos a comprovar o modo de como a aplicação do Lean IT que é capaz de aumentar a produtividade de forma duradoura e sustentável (Elbert et al., 2012). Esta dissertação pretende apresentar através de um caso de sucesso que é possível a aplicação dos princípios e práticas Lean ao IT do sector bancário e responder diretamente à questão orientadora: Pode o Lean IT contribuir para aumentar a produtividade das equipas?

Para este trabalho, tanto na componente de revisão de literatura como na produção do estudo de caso, recorreu-se a diferentes fontes de informação como documentos académicos e de suporte à atividade empresarial, além de relatos dos participantes e observadores diretos dos projetos relativos aos casos de estudo.

Esta dissertação desenvolve-se em três capítulos: o primeiro aborda os conceitos respeitantes ao domínio, o enquadramento histórico das metodologias tradicional e Agile, as suas vantagens, desvantagens e principais lacunas. Trata também da temática respeitante ao *pensamento* Lean e à sua aplicação aos SI, terminado com a referência a dois famosos casos de sucesso da literatura (Ericsson e Wipro); o segundo procura explicitar o tipo de investigação e método de pesquisa utilizados para concretizar este trabalho; no terceiro capítulo apresentam-se dois casos de estudo reais da indústria dos SI, um para ilustrar o típico projeto tradicional e outro para dar a conhecer a implementação de uma iniciativa de gestão Lean IT no sector bancário.

Apresenta, ainda, um texto conclusivo acerca da investigação que a suporta, além de sugerir possíveis futuros trabalhos nesta área.

1 Estado da Arte

1.1 Modelo em Cascata

Por volta dos anos cinquenta do século XX, a engenharia de *software* ainda não tinha arrancado em moldes verdadeiramente profissionais, a sua implementação estava vinculada, em exclusivo, à produção do hardware destinado, sobretudo, na indústria aeroespacial e na defesa norte americana. Tratava-se, por isso, de uma engenharia cujo desempenho assentava nos mesmos princípios utilizados na produção de *hardware*, os quais tinham por máxima fundamental: ‘Medir duas vezes, cortar uma.’¹(Boehm, 2006).

A partir da década seguinte começou, porém, a perceber-se que a engenharia do *software* e a de hardware tinham diferenças substanciais (Boehm, 2006), distinguindo-se, por exemplo, pela circunstância de facilmente se poder modificar *software* ao contrário do que se observava em relação ao hardware. Ora a possibilidade de agir sobre o *software* sem grandes constrangimentos abriu caminho a novas formas de abordagem da respetiva produção, ganhando terreno, nomeadamente, a prática ‘*code and fix*’ (programar e corrigir) também conhecida como Modelo *Ad Hoc* (Connors, 1992 e Boehm 2006). Nesta abordagem o sistema era concebido sem especificação ou desenho, sendo modificado até atingir a satisfação do utilizador o que originava muito código esparguete e sistemas de difícil manutenção.

Data de finais da década de sessenta do século XX as primeiras conferências internacionais de vulto sobre Engenharia de Software, as quais se realizaram em 1968 e 1968 pelo Comité de Ciências da NATO, cujo objetivo era discutir os problemas do ramo (Somerville, 2011) e nelas se concluindo pela necessidade de adoção de métodos mais organizados e de práticas mais disciplinadas de forma a se poder escalar os projetos. (Boehm, 2006)

No processo de procura de novos métodos de gestão da produção, ainda na década de setenta Royce (Winston Royce, engenheiro de software americano) viria a contestar o ‘*code and fix*’, alegando que os processos de produção deviam envolver uma

¹ Provérbio que pretende frisar a ideia que é preferível perder tempo a verificar do que remediar depois de iniciar.

programação cuidadosamente organizada, a qual deveria ser precedida por um desenho de *software* assente no desenvolvimento prévio da engenharia de requisitos. Nascia assim o modelo em cascata que passou a ser o mais usado em grandes projetos e de elevada complexidade. (Ji & Sedano, 2011)

Com particular ênfase no planeamento efetuado no período inicial do projeto (Miguel, 2015), o modelo em cascata de desenvolvimento de *software* ganhou tal designação em virtude das diferentes fases de execução que comporta ocorrerem de forma sequencial (ver figura 1). A prioridade atribuída pelo modelo a um rigoroso e detalhado planeamento na fase de arranque pretende prevenir a ocorrência de eventuais falhas de *design* antes do desenvolvimento/implementação respetivos, assumindo-se como especialmente relevante para projetos em que o controlo de qualidade constitui uma das principais preocupações (Balaji & Murugaiyan, 2012).

O ciclo de vida em cascata puro integra várias fases nunca sobreponíveis, arrancando com o estabelecimento de requisitos de sistema e requisitos de *software*, a que se seguem o desenho da arquitetura, o desenho detalhado, a codificação, os testes e a manutenção.

Dada a relevância que entretanto ganhou, o modelo em cascata serviu e ainda serve como base a muitos outros modelos de desenvolvimento de ciclo de vida de software (SDLC – *System Development Lifecycle*) (Munassar & Govardhan, 2010)

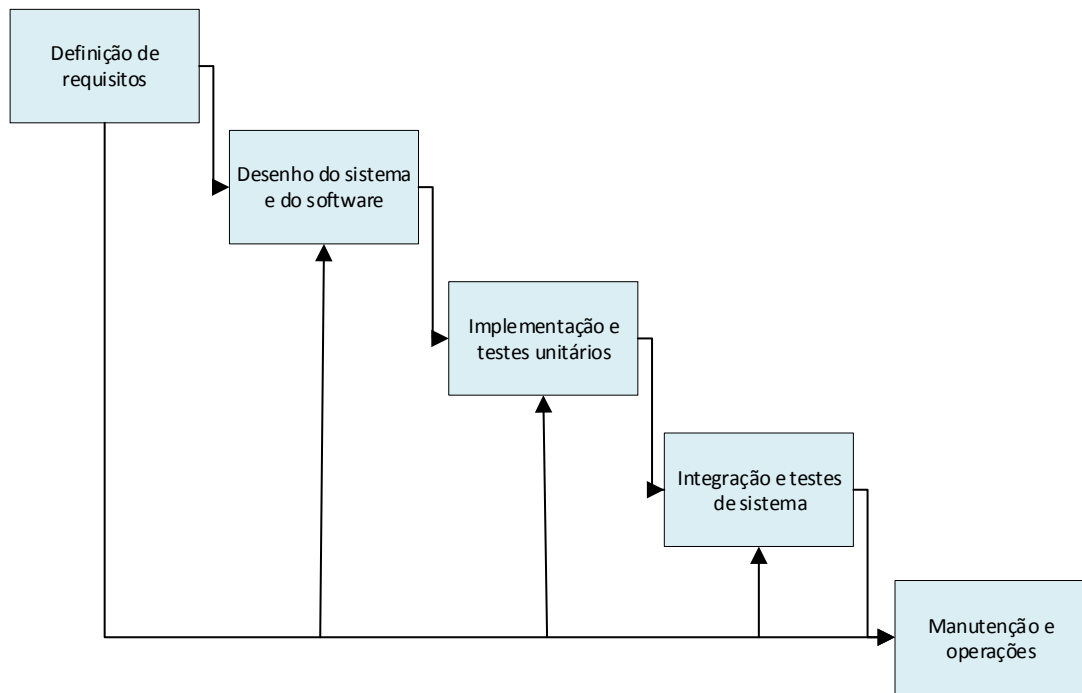


Figura 1: Desenho esquemático do modelo de desenvolvimento em cascata (Sommerville, 2011)

As diferentes fases sequenciais que integram o modelo em apreço podem descrever-se resumidamente do seguinte modo:

Definição dos requisitos: estabelecimento das funcionalidades e expectativas para as funcionalidades do *software* a desenvolver. Definem-se também os requisitos do sistema que novo *software* irá afetar. A análise dos requisitos implica a identificação da interação que será necessária com outras aplicações e bases de dados, requisitos de performance, requisitos de interação com o utilizador. (Munassar & Govardhan, 2010)

Desenho do sistema e do software: desenho da arquitetura global resultante da agregação dos requisitos de *hardware* necessários bem como os de *software*. (Sommerville, 2011). Identificação e detalhe das abstrações fundamentais para o sistema de *software* e bem como o levantamento das suas relações.

Implementação e testes unitários: nesta fase, o que anteriormente foi desenhado, é codificado num conjunto de programas. Os testes unitários consistem na verificação individual que cada função atende às especificações pretendidas. (Sommerville, 2011).

Integração e testes de sistema: cada porção de *software* ou unidade é incorporada num sistema completo onde é testada de forma a garantir que o sistema funciona como um todo e que cumpre com o definido. Assim que terminam os testes integrados, o sistema de *software* segue para validação do cliente. (Sommerville, 2011).

Manutenção e operações: nesta que é geralmente a fase mais longa do projeto procede-se à instalação e arranque do sistema. A operação de manutenção, que também inclui, implica a correção de erros não identificados descobertos em fases anteriores, ou, inclusivamente, a introdução de melhorias às unidades já implementadas de forma a satisfazer novos requisitos entretanto surgidos (Sommerville, 2011).

A existência de documentação detalhada em todas as fases do processo representa uma característica transversal e que define este modelo, que tem como outras das suas orientações realização de uma rigorosa avaliação no final de cada etapa do projeto por forma a aferir-se da possibilidade de progressão à fase seguinte (Sommerville, 2011).

1.1.1 Pontos fortes do modelo

Uma vez que assenta, como se referiu anteriormente, no pressuposto de um planeamento prévio das diferentes fases, o modelo em cascata apresenta como uma das suas principais vantagens o fato do plano do projeto estar completamente definido logo no seu início. A definição deste plano é efetuada com base nos requisitos identificados, o que baixa o risco e o custo do projeto relativamente ao âmbito (Miguel, 2015). Além disso, a definição do plano implica a identificação dos entregáveis, os momentos de contacto com os *stakeholders*, as milestones e inclusive as atividades de desenvolvimento (Fernandez, D. & Fernandez, J. 2008, Hass, 2007 e Munassar & Govardhan, 2010) e (PMI, 2012).

Trata-se igualmente de um modelo que enfatiza a definição de requisitos (Miguel, 2015), o que faz com que esta seja uma atividade que decorra cedo no projeto,

de máxima prioridade, e que é executada com muito detalhe (Fernandez, D. & Fernandez, J. 2008, Hass, 2007)

Uma vez que os requisitos são detalhados na fase de arranque do projeto, os recursos pertencentes às equipas de desenvolvimento não têm de ser os melhor qualificados (Fernandez, D. & Fernandez, J. 2008), podendo estes estar dispersos em localizações geográficas diferentes pois o âmbito está completamente fechado. Desta forma, o modelo em cascata assume-se como metodologia ideal para aplicação em produtos bem conhecidos e maduros (Munassar & Govardhan, 2010).

Além disso, pode dizer-se que o modelo em cascata é de fácil entendimento e implementação, o que tem contribuído para a sua divulgação internacional, traduzindo-se na ajuda à implementação das boas práticas na área de engenharia de *software* - primeiro definir, só depois desenhar e implementar. Mesmo as equipas de desenvolvimento pouco experientes facilmente conseguem adaptar-se ao modelo, aprender com ele e adotá-lo. (Munassar & Govardhan, 2010)

1.1.2 Pontos fracos do modelo

Entre as lacunas evidenciadas pelo modelo, figura, em contrapartida, a circunstância de, em determinados casos, se revelar a impossível a definição detalhada e completa dos requisitos nas primeiras fases do projeto (Munassar & Govardhan, 2010). Ao postular a definição nas primeiras fases do processo do plano e cronograma do projeto, o modelo em cascata falha quanto não há, à partida, uma visão completa e detalhada dos requisitos, não acomodando facilmente alterações. Além disso as mudanças de requisitos são muito dispendiosas de implementar, as iterações requerem muito esforço visto obrigarem a refazer parte do trabalho (Fernandez, D. & Fernandez, J. 2008 e Hass, 2007). Uma das explicações para o aumento do custo é a disponibilização tardia do *software* para validação, o que traz mais impacto se a deteção de defeitos ocorresse numa fase inicial (Munassar & Govardhan, 2010). Acresce o facto de ser exigido a aprovação de documentos em cada etapa, o que torna a processo moroso e conseqüentemente mais dispendioso. Resumidamente, o modelo em cascata está relacionado com custos e esforços elevados (Fernandez, D. & Fernandez, J., 2008), (Hass, 2007) (Petersen, Wohlin & Baca, 2009).

Os problemas que ocorrem nas várias fases sequenciais do projeto, quando não resolvidos são empurrados para as fases posteriores do processo. Segundo a literatura (Petersen *et al.*, 2009) a principal falha de projetos em grande escala que utilizam o modelo em cascata, resulta da impossibilidade de gerir os requisitos de forma bem-sucedida. Consequentemente, os clientes não sentem que as suas necessidades atuais estejam a ser endereçadas e o *software*, aquando da sua entrega irá conter várias funcionalidades que não serão utilizadas e no entanto foram implementadas de acordo com o plano. (Petersen *et al.*, 2009)

Outra das desvantagens do modelo assenta na dificuldade que revela em potenciar melhorias de produtividade e aumentar a frequência de desenvolvimento e de entrega do produto ao cliente. O grande foco do modelo será sempre entregar de acordo com o planeamento deixando para segundo plano a entrega de valor ao cliente. (Fernandez, D. & Fernandez, J. 2008)

Resumo dos problemas identificados no modelo em cascata: (Petersen *et al.*, 2009)

Tabela 1: Listagem dos problemas encontrados na metodologia em cascata

ID	Problema
1	Esforço e custo elevado na execução de documentos detalhados e necessidade de aprovação antes da fase de desenvolvimento.
2	Mudanças de difícil acomodação
3	Uma iteração sobre uma determinada fase requer um grande esforço de retrabalho.
4	Quando finalmente o sistema é utilizado pelo cliente este descobre problemas que surgiram nas fases iniciais. Quando faz esta avaliação é efetuada o sistema não reflete o estado atual dos requisitos.
5	Os problemas não resolvidos das fases sequenciais já terminadas são deixados para as fases posteriores.
6	Integração <i>big-bang</i> e o teste dos sistemas como um todo no fim do projeto podem levar a problemas inesperados de qualidade, custos elevados e podem fazer com que se ultrapasse o cronograma estipulado.
7	Falta de oportunidade para o cliente dar o seu feedback sobre o sistema
8	Este modelo aumenta o tempo de espera devido à grande quantidade de artefactos de <i>software</i> que tem de ser aprovados em cada etapa.

São raras as vezes em que os projetos seguem o fluxo sequencial, ou a metodologia em cascata pura, uma vez que os clientes não conseguem especificar os requisitos com clareza e de forma completa na fase inicial do projeto (Hass, 2007).

Apesar de todos os pontos fracos, as muitas vantagens deste modelo de desenvolvimento garantem que se mantenha como uma das metodologias mais populares de desenvolvimento de *software* em todo o mundo (Balaji & Murugaiyan, 2012).

1.1.3 Gestão tradicional de projeto

Na década seguinte àquela em que a engenharia de *software* começava a dar os primeiros passos, isto é nos anos sessenta do século XX, o desenvolvimento da atividade determinou a necessidade de criação de uma entidade destinada a assegurar o reconhecimento da gestão de projetos. Para responder a essa exigência dos novos tempos surgiria em 1969, no Instituto de Tecnologia da Geórgia (Atlanta), o PMI (Project Management Institute), organização sem fins lucrativos que tinha por meta principal fomentar um fórum de debate sobre os problemas de gestão de projeto, soluções e possíveis aplicações. Caberia aliás ao PMI a iniciativa de lançamento, em 1996, da primeira versão do PMBOK® (Project Management Body Of Knowledge) (PMI, 2012), um guia de gestão de projetos, considerado a base do conhecimento pelos profissionais da área que o reconhecem como contendo as boas práticas da profissão. O guia, que já conheceu seis edições, procura responder à necessidade identificada nos anos noventa do século passado de documentar e padronizar práticas geralmente aceites pela comunidade.

Nos termos do PMBOK a gestão de projeto passa a ser encarada como ‘A aplicação de conhecimentos, aptidões, ferramentas e técnicas às atividades do projeto, com o objetivo de satisfazer os requisitos do projeto’ (PMI, 2012)

O guia editado pelo PMI descreve as práticas, os processos, fases e conhecimento das áreas comumente consideradas nos projetos, estabelecendo cinco grupos de processos (inicialização, planeamento, execução, monitorização e controlo, e encerramento) e 10 áreas de conhecimento (integração, âmbito, tempo, custo, qualidade,

recursos humanos, comunicação, risco e gestão de aquisições e Partes Interessadas) (Balaji & Murugaiyan, 2012, Miguel, 2015 e PMI 2012).

Apesar do PMBOK não especificar concretamente a metodologia de desenvolvimento recomendada como a adotada no modelo em cascata, a maioria dos gestores que seguem os processos do PMI acabam utilizando este procedimento tipicamente suportado por diagramas de Gantt (Sutherland & Ahmad, 2011, Balaji & Murugaiyan, 2012 e Sutherland, 2016)

A gestão de projeto tradicional, nos termos definidos pelo guia, envolve planeamento intencional deliberado, disciplinado e a utilização de métodos de controlo. Com esta abordagem, as diferentes fases de ciclo de vida do projeto podem ser facilmente identificadas, as tarefas podem ser terminadas uma após outra numa sequência definida, o que requer que parte do projeto já esteja planeado de forma antecipada. (Hass, 2007) e (Sutherland & Ahmad, 2011).

Resumidamente, a gestão de projeto tradicional possui as seguintes características:

Faseada. A gestão de projeto tradicional é dividida em fases com atividades homogéneas distintas, como os requisitos, o desenho, implementação, verificação e fase de manutenção. São executadas passagens de testemunho entre fases e isso impulsiona o projeto para a fase seguinte.

Sequencial. As fases de um projeto são tipicamente sequenciais, onde uma fase só se inicia quando a anterior está finalizada e bem-acabada.

Não-iterativa: As fases do projeto não são tipicamente repetidas, a não ser que haja um processo formal com controlo de mudança que possa desencadear uma nova execução das fases e de forma sequencial.

Orientada ao plano: é orientada a um plano de projeto bem detalhado que descreve várias atividades durante as fases e prediz a cronologia das atividades, tarefas, fases de projeto e o esforço que será necessário para executá-los.

Os gestores tradicionais gerem o projeto contra o orçamento, cronograma e âmbito (Fernandez, D. & Fernandez, J., 2008 e Sutherland & Ahmad, 2011). As métricas e a variância podem ser monitorizadas *versus* a base que foi inicialmente estabelecida e planeada por estes. O gestor tradicional pretende reduzir e responder aos

riscos, mantendo inclusive as restrições de tempo e orçamento. (Fernandez, D. & Fernandez, J., 2008). A manutenção do plano também é da sua responsabilidade, bem como a atualização dos diagramas de Gantt e produção atempada de relatórios de projeto com métricas rigorosas (Sutherland & Ahmad, 2011).

O PMBOK descreve ainda o gestor como o maior *stakeholder* do projeto, responsabilizando-o por comunicar com os restantes *stakeholders* e atribuindo-lhe o papel de principal patrocinador do projeto, equipa, bem como dos outros elementos chave que contribuem para o sucesso do empreendimento (Sutherland & Ahmad, 2011 e PMI, 2012).

O gestor de projeto tradicional ocupa, assim, o centro das interações entre os *stakeholders* e o próprio projeto (Sutherland & Ahmad, 2011 e PMI, 2012), garantindo excecional importância ao seguimento das atividades de acordo com o plano previamente aprovado pelas partes interessadas.

Atualmente, porém, os processos de negócio, cada vez mais complexos, interconectados, dependentes e mais interrelacionados do que nunca. (Hass, 2007), tendem a rejeitar estruturas organizacionais tradicionais para poderem estabelecer comunidades complexas constituídas por alianças estratégicas com fornecedores, redes de clientes, vendedores *outsourcing* e parceiros chave provenientes de diferentes grupos, entidades regulatórias e ainda a concorrência.

Através desses acordos, as organizações conseguem endereçar a pressão da mudança, a competição global, a diminuição do *time-to-market*, a mudança rápida das tecnologias e o aumento de complexidade a cada *milestone*. Devido à natureza multifacetada dos negócios, os projetos que implementam novos sistemas de negócio também são mais complexos (Hass, 2007).

Em resultado das exigências dos processos atuais a gestão tradicional assenta em pressupostos que estão desfasados das necessidades atuais e não consegue acomodar facilmente a natureza iterativa e o desenvolvimento exploratório (Munassar & Govardhan, 2010) pelo que deve ser encontrada uma solução mais adequada aos desafios modernos.

1.2 Modelos de Maturidade IT: CMMI e ITIL

1.2.1 CMMI

O Departamento de Defesa norte-americano apercebeu-se, durante a segunda metade da década de 1980, de que grande parte dos sistemas informáticos desenvolvidos para aplicação na área militar se estavam a tornar pesados em termos de *software*, não indo ao encontro dos requisitos estabelecidos, além de implicarem custos incontroláveis. Face a essa constatação, decidiu solicitar à *Carnegie Mellon University* um estudo patrocinado pelo Governo para lançamento de um guia de práticas na área de engenharia da especialidade tendente à melhoria da qualidade dos sistemas dependentes de *software* (Kasse, 2004). Tal iniciativa resultaria no aparecimento, em 1991, do CMM® para *software* v1.0. Mais tarde surgiria uma versão já integrada, o CMMI®, considerada uma abordagem aperfeiçoada que faculta às organizações os elementos essenciais de implementação de processos eficientes (Software Engineering Process Group, 2008). Para o Departamento da Defesa, como para o exército, empresas governamentais e grandes corporações assumem-se como os fatores decisivos no que ao *software* diz respeito a confidencialidade, a segurança e a estabilidade, os quais se sobrepõem a *items* como preço ou a capacidade de adaptação à mudança de necessidade dos clientes. (Ziółhowski & Deregowski, 2014).

Nesse contexto, o CMMI avalia e atribui um grau de maturidade aos processos de uma organização utilizando uma escala de um a cinco, sendo um considerado o nível 5 o mais maduro. (Hibbs et al. 2009).

As organizações classificadas como menos maduras pela avaliação CMMI podem apresentar características como a improvisação de processos durante o decorrer dos projetos, desrespeito pelo seguimento de processos anteriormente aprovados, postura tendencialmente reativa em vez de proactiva, planeamento e orçamentos irrealistas com implicações ao nível da necessidade de sacrificar a qualidade dos produtos em prol do plano.

Pelo contrário, as organizações consideradas maduras apresentam uma boa comunicação e coordenação entre os membros do grupo de trabalho e os seus objetivos

são atingidos conforme o plano. (Hibbs et al. 2009). A implementação do CMMI traz vantagens significativas, como ganho de uma visão holística que não se concentra apenas nas partes específicas do negócio, tratando a organização como um todo coerente, além de tentar endereçar todos os problemas tradicionalmente enfrentados pelas organizações IT. (Ziólowski & Deregowski, 2014).

1.2.2 ITIL

Desenvolvido pela *Central Computer and Telecommunications Agency* (CCTA) e inicialmente concebido para o Governo britânico na década de 1980, o ITIL tinha por objetivo estabelecer o conjunto de melhores práticas a implementar nos centros de informação governamentais. (ITIL Portugal, 2017). Enquanto conjunto de bibliotecas de domínio público com vista a dar suporte aos serviços de IT, tem sido desde a sua criação adaptado e revisto para ser utilizado em várias indústrias (Guedes, 2011).

Segundo Davis & Bentley (2010), o ITIL é uma *framework* customizável constituída por princípios e técnicas para gerir infraestruturas em IT, o desenvolvimento e operações. Nos seus cinco livros descreve práticas importantes e prescreve *checklists*, tarefas e procedimentos que podem ser aplicados em qualquer empresa de IT. Assim, a *framework* promove a integração do IT com o negócio, portfólio de serviços dinâmico e providencia as melhores práticas de gestão do ciclo de vida do serviço (SEPG, 2008). Esta *framework* permite que gestores de organizações de IT caóticas implementem vários processos passo-a-passo com a confiança de que poderão ter melhorias significativas (Addy, 2007).

Devido à sua natureza prescritiva, o ITIL não requer muito esforço de reflexão e decisão sobre procedimentos, desta forma e como já está tudo detalhado não há necessidade de perder tempo com esta tarefa (Addy, 2007).

Outro aspeto positivo prende-se com o facto de promover a especialização de funções em IT, atribuindo funções e responsabilidade que podem ajudar a aumentar a retenção de recursos humanos nas organizações, encorajados pela possibilidade de terem uma longa carreira bem definida e especializada. (Addy, 2007).

Tanto o CMMI como o ITIL promovem a melhoria dos processos de desenvolvimento de *software* e a sua qualidade, o que se traduz numa redução do custo associado à qualidade (SEPG, 2008).

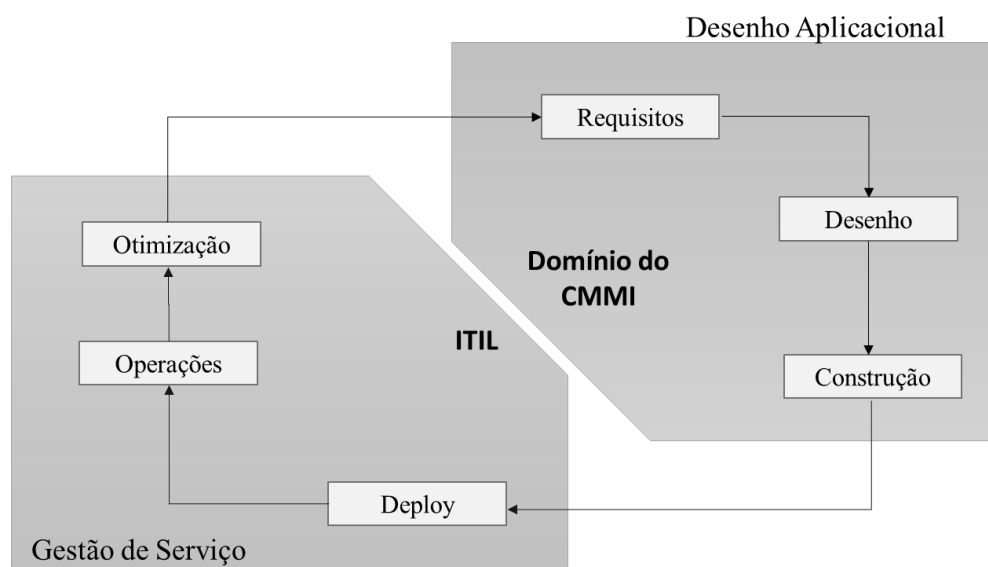


Figura 2: Comparação ITIL e CMMI (Adaptado de SEPG, 2008)

Tanto o CMMI quanto o ITIL têm *frameworks* de maturidade de processos que seguem uma abordagem similar e estruturada. Ambos enfatizam o desenvolvimento de processos para melhorar o desenvolvimento de produtos e assegurar a satisfação do cliente, estimulando a coordenação de atividades multidisciplinares relacionadas com um projeto.

Frequentemente as empresas deparam-se com dificuldades de implementação, acabando por considerar o CMMI ou o ITIL abordagens complexas, dada a existência de numerosas regras a cumprir e processos que devem ser seguidos em conformidade de acordo com o estabelecido. (SEPG, 2008). Há esforço despendido na implementação, grande consumo de tempo e muitas vezes é necessária uma mudança na cultura de empresa (Ziółhowski & Deregowski, 2014).

Para se garantir sucesso na implantação destas *frameworks* torna-se necessário haver um comprometimento por parte gestão, que deve verificar o estado de todos os processos executados pelas equipas, o que requer dispêndio de tempo e esforço suficiente para compreender o ponto de situação atual, fazer o mapeamento para o estado futuro das operações e focar-se profundamente no negócio enfrentando os seus problemas de forma comprometida (Addy, 2007). Críticos destas *frameworks* enquadram-nas geralmente na engenharia clássica em que os modelos são similares aos de uma linha de montagem e promovem os processos em vez da substância, o que

consideram que atrasa o *time-to-market* (Shelton, 2008). Contudo, quando aplicados a uma organização, o ITIL e CMMI, *standards de facto*, podem traduzir-se, de forma genuína, numa mais-valia para a melhoria dos seus processos. A única preocupação que lhes está associada refere-se ao cumprimento de ‘*checklists*’ para obtenção do título ou do *rating* então o resultado é apenas mais um processo pesado, burocrático e pouco benéfico. (Hibbs et al. 2009).

1.3 Agile

O incremento exponencial da introdução de computadores nas empresas, de que resultará numa elevada procura de *software* vai criar, o início dos anos noventa do século XX, um outro problema ao setor, a chamada “crise de desenvolvimento aplicacional” ou “crise do atraso na entrega aplicacional”. Os especialistas industriais estimavam que então eram necessários cerca de três anos entre a validação das necessidades de negócio e a entrada em produção das aplicações requeridas. Tratava-se de um ritmo de produção de *software* bastante aquém do imposto pelas necessidades das empresas, sendo mesmo inferior às necessidades empresariais observadas há 25 anos atrás. Acresce que a demora entre a encomenda de aplicações e a sua entrada em funcionamento ultrapassava, em muitos casos, os três anos referidos. Chegavam a registar-se situações em que eram desenhados, desenvolvidos e entregues sistemas de *software* e hardware bastante complexos num intervalo de tempo que podia prolongar-se por décadas. Uma tal delonga começou a gerar frustração entre os líderes da indústria de *software* e, segundo o engenheiro aeroespacial norte-americano Jon Kern, já na década de 90 se estava “à procura de algo que fosse mais oportuno e responsivo” (CITAÇÃO). Para encontrar soluções visando ultrapassar práticas aparentemente pouco produtivas, que trouxeram grande insatisfação (Sommerville, 2011) aos utilizadores, Kern e outros 16 criadores *software* e representantes do setor reuniram-se em 2001, em Snowbird, Utah, Estados Unidos, decididos a propor “métodos mais ágeis e leves”. Do encontro resultaria o Manifesto Agile que se traduziu no estabelecimento de um conjunto de valores e princípios nucleares de suporte a equipas de alta performance da indústria de *software* e que serão brevemente descritos na secção 1.3.1.

Numa apreciação às conclusões a que chegou a reunião de Utah, Sutherland (2013), um dos presentes no encontro de Utah, garante que as orientações adotadas no manifesto de Agile não constituem uma metodologia *per si*, correspondendo antes a um termo *umbrela*, pois inclui na sua alçada várias metodologias ágeis. Tais metodologias ágeis como SCRUM, ASD, Crystal Methods, DSDM, XP, FDD teriam já começado a surgir no final da década de 1990 (Larman & Basili, 2003), mas só agora surgiam unificadas e fundamentadas no Manifesto. Por exemplo, o SCRUM, apresentado como processo formal numa conferência de OOPSLA em 1995 (Sutherland e Ahmad, 2011 e Azanha et al., 2017), ganhou o estatuto de uma das metodologias Agile mais simples e comuns (Sutherland & Ahmad, 2011). O XP, metodologia igualmente bastante popular, começou a ser aplicado em 1996, por Beck no seu projeto C3², sendo mais tarde (1999) detalhado no livro *Extreme Programming Explained – Embrace Change*. Quer o SCRUM quer o XP afirmaram-se como as metodologias ágeis aplicadas em todo o mundo (Hoda, 2012) com cerca de 80% das implementações (Sutherland, 2013), o que explica a abrangência registada no estudo dos processos nucleares e respetivas práticas

O SCRUM, considerado uma *framework* para processos de desenvolvimento e por isso destituído de práticas de engenharia específicas, foi especialmente desenvolvido para lidar com a mudança de requisitos. A designação SCRUM tem que ver com o rugby inglês, desporto onde os membros da equipa, coesamente unidos, jogam a bola na direção pretendida, protegendo a sua posse. Na metodologia, o SCRUM movimenta o projeto para a frente, melhorando a comunicação entre os membros da equipa e dividindo o trabalho em pequenas partes (designadas de *sprints*), as quais não devem prolongar-se além de quatro semanas. O SCRUM foca-se mais na gestão de processos do que nas técnicas de programação de *software* (Livermore, 2008), enquanto o XP possui técnicas ausentes no SCRUM (Sutherland, 2013), como conceitos focados no desenvolvimento do código, programação em pares para aumento da qualidade, desenho de testes anterior ao desenvolvimento, refatorização, propriedade coletiva e integração do código, pequenas *releases*, reuniões de pé, prevendo, inclusive, 40 horas de trabalho semanal, de modo a minimizar a fadiga e a perda de perspectiva das equipas (Livermore, 2008).

² Chrysler Comprehensive Compensation System sistema de pagamentos.

Segundo (Sutherland, 2013) as melhores equipas de implementação utilizam o SCRUM, recorrendo ao mesmo tempo das técnicas do XP. O SCRUM ajuda o XP a escalar melhor e o XP ajuda o SCRUM a ter um melhor desempenho.

Na perspetiva dos defensores das metodologias ágeis, os processos que preconizam revelam-se mais adaptativos do que preditivos e mais orientadas às pessoas, relativamente aos tradicionais (Abdulla, Holcombe & Gheorge, 2006). A abordagem adaptativa afigura-se como a melhor opção quando se tem por principal objetivo ajudar as organizações a ir ao encontro dos atuais desafios da economia digital (Maruping *et al.*, 2009). São eles (?), de resto, que desencadeiam a constante mudança, trazendo requisitos incertos e voláteis (Abdullah *et al.* 2006, Boehm & Turner 2005 e Boehm 2002).

Nesse contexto, Fernandez, D. & Fernandez, J. (2008) argumenta que o Agile assume o carácter de uma forma de pensar a solução de *software*, que passa a ser encarada como conjunto de pequenos entregáveis incrementados com o envolvimento do cliente ao longo das diferentes fases do seu desenvolvimento. Boehm (2005), sublinha, por seu lado, a dimensão iterativa desse desenvolvimento, graças à integração contínua do código. Fernandez, D. & Fernandez, J. (2008) acrescenta ainda que um dos principais objetivos do Agile é ter zero defeitos pelo que proporciona e defende um grande envolvimento da área de garantia de qualidade em cada ponto, como ilustra a Figura 3: Ciclo de vida de um projeto Agile - Adaptado Hass (2007).

Já Maruping *et al.* (2009) argumenta que um conjunto de técnicas flexíveis, as metodologias ágeis, permitem capacitar as equipas de desenvolvimento de *software* para enfrentar os desafios dos tempos modernos, adicionando flexibilidade aos projetos e assegurando às equipas possibilidades para trabalharem mais eficazmente. Boehm (2002) acrescenta, por seu turno, que os métodos ágeis adquirem toda a sua agilidade graças à confiança que depositam no conhecimento tácito dos membros das equipas em vez de documentação com detalhe dos planos de execução de tarefas.

Para Stellman & Greene (2015) entre as principais vantagens do Agile, uma metodologia que classifica de “emergente”, figura a liberdade garantida às pessoas que a implementam, especialmente no que se refere à decisão sobre o que deve ser aplicado a cada situação com base na sua análise empírica, orientando as suas ações pelos princípios e valores constantes do gráfico abaixo e que a seguir se especificam:

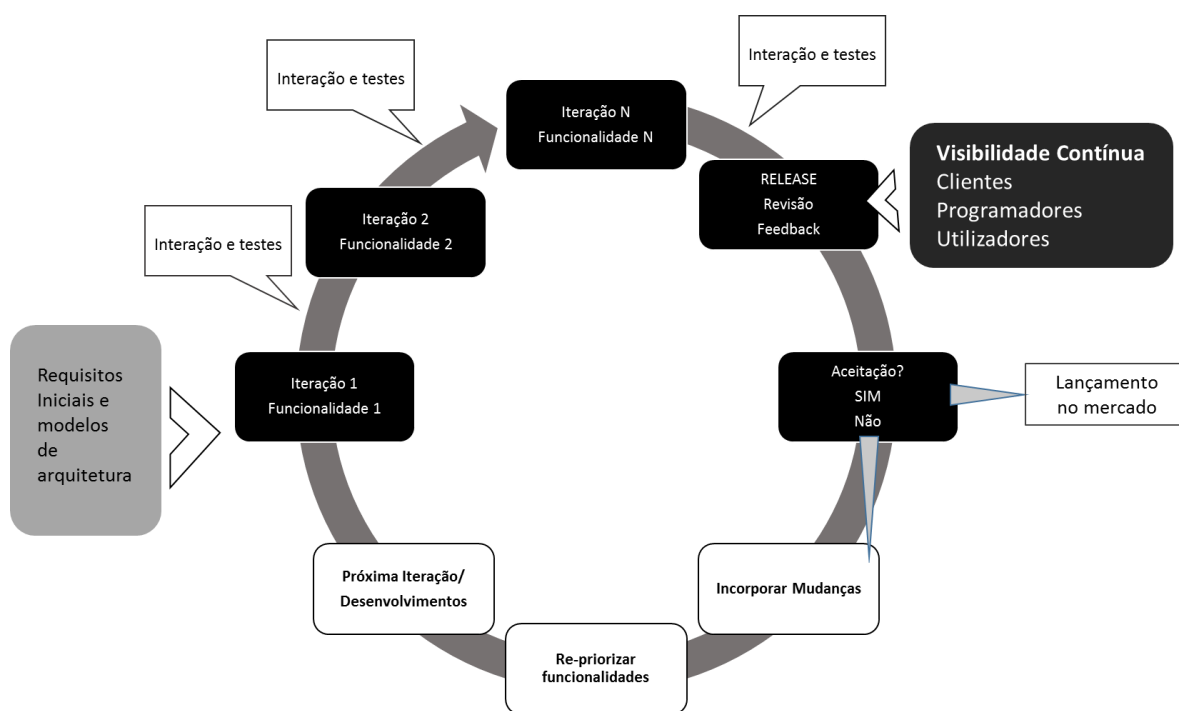


Figura 3: Ciclo de vida de um projeto Agile - Adaptado Hass (2007)

1.3.1 Valores e princípios Agile

São quatro os valores destacados no Manifesto Agile, cabendo a cada metodologia a decisão de aplicá-los conforme sua própria estratégia, sem, contudo, ignorar as orientações necessárias ao desenvolvimento e entrega de *software* funcional de elevada qualidade.

O documento inscreve como valores essenciais a consideração das pessoas como entidades superiores aos processos e ferramentas, a colocação dos produtos funcionais acima da documentação excessiva, o entendimento do trabalho colaborativo com os clientes mais importante do que as negociações com eles estabelecidas e a procura de resposta às mudanças em vez do seguimento de um plano pré-estabelecido (Beck, Beedle & Bennekum, 2001).

Princípios

No que se refere a princípios, o documento inúmera um total de doze, apontados como guia para metodologias incluídas no chapéu do ‘Movimento Agile’,

considerado como alicerce de uma cultura recetiva à mudança e com foco prioritário no cliente. Resumidamente são os seguintes os princípios em referência:

Princípio 1: Valor

“Garantir a satisfação do cliente, através da entrega rápida e contínua de software funcional”
(Beck et al., 2001)

O valor adicional do trabalho das equipas ágeis não reside na entrega de um produto no final de um projeto, consistindo, antes, na disponibilização de uma solução que se traduza em melhores resultados para o negócio do cliente (Cobb, 2011). Para a obtenção de tal meta, a equipa prioriza, com base na satisfação do cliente, a entrega da solução por funcionalidades tendo em conta as necessidades detetadas no dia-a-dia. As equipas ágeis usam, assim, uma abordagem baseada no valor e risco para determinar o que deve ser implementado e consequentemente planeado em primeiro lugar. Inclusivamente na definição do plano é tido em conta o valor associado a cada tarefa, não se perdendo de vista os interesses do cliente. Uma tal estratégia torna necessário decidir qual o esforço mínimo e essencial para definir e planear o projeto, determinar qual o risco associado ao ter parte do planeamento incompleto e estar ciente do valor ganho quando parte é planeado com antecedência. Este tipo de decisão deve ser tomada conjuntamente entre os patrocinadores e as partes interessadas do projeto.

Princípio 2: Flexibilidade

“Até mesmo mudanças tardias ao âmbito do projeto são bem-vindas.” (Beck et al., 2001)

Para que o cliente possa retirar vantagens competitivas, os processos ágeis devem adequar-se às mudanças, sendo precisamente a este tópico que se refere o princípio em causa. Num mercado cada vez mais competitivo e complexo é necessário desenvolver projetos de forma flexível para que estejam abertos a mudanças, em qualquer altura e mesmo no fim do desenvolvimento. Toda a ênfase atribuída à flexibilidade no projeto resulta exclusivamente da necessidade de atender à satisfação dos interesses do cliente já inscrito como prioridade no primeiro princípio.

Princípio 3: Frequência

“Software funcional é entregue frequentemente;” (Beck et al., 2001)

Quando se fala em metodologia Agile, fala-se em desenvolvimento incremental, o que implica entregas frequentes de *software* passível de ser utilizado pelo cliente, que, ao receber o incremento, executa um exercício de valor sobre ele, circunstância que permite o estabelecimento de um alinhamento de expectativas das partes envolvidas.

A divisão das funcionalidades em *slots* para iterações que terminam com uma entrega, favorece também a priorização dos requisitos, endereçando as funcionalidades mais importantes em primeiro lugar. Relativamente ao plano de iterações, a metodologia considera preferível adiar o máximo possível o planeamento de cada *sprint* e utilizar o planeamento *just-in-time* sempre que praticável e à medida que o projeto progride.

Princípio 4: União

“Cooperação constante entre as pessoas que entendem do ‘negócio’ e os programadores”
(Beck et al., 2001)

As metodologias ágeis baseiam-se fortemente no espírito colaborativo entre os indivíduos participantes, nomeadamente na componente de colaboração com o cliente. Espera-se que o utilizador do *software* participe ativamente na solução, estando presente para definir e gerir os requisitos do projeto de modo a sentir-se tão responsável quanto a restante equipa pelo sucesso do empreendimento comum.

Princípio 5: Motivação

“Os projetos devem ser criados em torno de indivíduos motivados. Dê-lhes o ambiente e o apoio que necessitam, e confie-os para começar o trabalho feito.” (Beck et al., 2001)

O manifesto refere como fundamental que a equipa de projeto esteja motivada para que desempenhe o seu papel em plenitude, considerando que o ambiente deve ser adequado e propício para a execução das atividades de desenvolvimento. Insiste na importância do gestor ter bem presente o intento de atuar como facilitador sempre que necessário e a manter a equipa motivada através de garantia de um ambiente favorável, com o suporte exigido.

Princípio 6: Comunicação

“O método mais eficiente e eficaz de transmitir informações para dentro de uma equipa de desenvolvimento é a conversa cara-a-cara” (Beck et al., 2001)

Ao contrário das metodologias tradicionais, consideradas demasiado burocráticas, o Agile pretende melhorar a forma de comunicação por via da otimização da documentação produzida, que deve ser orientada ao valor e limitada ao essencial. Em vez da folha do registo de tarefas, as reuniões de planeamento das iterações revelam-se mais eficientes, segundo sugere, pois permitem a troca de conhecimentos frente-a-frente entre os vários membros da equipa, promovendo maior rapidez no alinhamento de todos em prol do objetivo comum.

Princípio 7: Funcionalidade

“Software funcional é a principal medida de progresso do projeto” (Beck et al., 2001)

Para o movimento Agile, um *software* funcional é por si a prova primária do progresso do projeto e não o somatório de várias tarefas terminadas, segundo o enfoque primário das metodologias tradicionais. O cerne do trabalho deverá, por isso, privilegiar sempre o ato de executar/desenvolver (*software*), seguindo as melhores práticas, e que será utilizado como medida de progresso.

Princípio 8: Sustentabilidade

“Os processos ágeis promovem o desenvolvimento sustentável. Os patrocinadores, programadores e utilizadores devem ser capazes de manter um ritmo constante indefinidamente.” (Beck et al., 2001)

O ambiente para o desenvolvimento de projeto, o plano, as iterações e o envolvimento das partes interessadas devem ser organizadas e geridas para que não se esgote nenhum dos recursos e se permita que o processo possa ser contínuo, sem interrupções, onde todos estejam disponíveis, quando necessário, para participar e dar o devido suporte. É esperado que o desenvolvimento possa ser sustentável de modo a não consumir os recursos humanos e, pelo contrário, possa multiplicar as suas capacidades para usufruto futuro.

Princípio 9: Revisão

“A atenção contínua à excelência técnica e ao bom design aumenta a agilidade.” (Beck et al., 2001)

Verificar significa conferir se a solução está de acordo com os requisitos detalhados na documentação do projeto, enquanto que validar quer dizer ver se a solução está de acordo com o pretendido pelo cliente final.

Constatar que um produto não satisfaz a necessidade do utilizador no fim da *release*³ pode trazer consequências muito negativas. Por isso as metodologias ágeis, para tentarem evitar o problema, garantem ênfase à validação do produto, revisão dos requisitos técnicos e de desenho, em vários pontos do ciclo de vida do seu desenvolvimento para que se possa entregar uma solução realmente alinhada aos objetivos do negócio do cliente.

Princípio 10: Simplicidade

“Simplicidade – a arte de maximizar a quantidade de trabalho não feito – é essencial.” (Beck et al., 2001)

Simplificar o trabalho de forma a não comprometer o tempo da equipa com registos e documentos dispensáveis e que não são uma mais-valia para o projeto. Sem a produção e análise de uma grande número de artefactos a equipa fica com o trabalho simplificado, o que leva à sua conclusão antecipada garantindo-se o *time-to-market*.

Princípio 11. Organização

“As melhores arquiteturas, requisitos e projetos emergem de equipas auto-organizadas.” (Beck et al., 2001)

A auto-organização da equipa e sua capacitação para encarar a solução em desenvolvimento como propriedade coletiva constitui característica proferida pelos defensores dos métodos Agile. Na sua perspetiva, a responsabilização dos membros das

³ Lançamento, entrega

equipas auto-organizados afigura-se crucial para que se possam assumir as melhores escolhas tanto a nível da arquitetura, como em termos de requisitos e do desenho necessários à solução a atingir.

Princípio 12: Autoavaliação

“Em intervalos regulares, a equipa reflete sobre como tornar-se mais eficaz, em seguida, ajusta o seu comportamento em conformidade” (Beck et al., 2001)

Detetar problemas e falhas precocemente assume particular relevância na produção de *software*, constituindo uma das razões pelas quais a abordagem Agile divide o projeto em pequenas iterações como meio de promover validações frequentes, recorrendo ao uso de testes automáticos à medida que ele evolui. O facto do ciclos de desenvolvimento serem curtos permite que se chegue ao seu fim com brevidade e haja a possibilidade de aprender com aquilo que correu menos bem e introduzir melhorias na iteração seguinte do ciclo. A estratégia de não ter esperar pelo fim de um longo projeto para analisar e incorporar as lições aprendidas torna esta metodologia bastante flexível, adaptativa e aberta às mudanças logo desde do início do projeto.

1.3.2 Gestão de Projeto Agile

“O trabalho de energizar, fortalecer e tornar equipas de projeto rápidas a entregar valor de negócio confiável em colaboração com o cliente resultando numa aprendizagem contínua e de adaptação às suas mudanças de necessidade e ambientais” (Augustine⁴, 2005, sobre a gestão de projeto em metodologia Agile)

Sabe-se que as empresas com uma boa base de clientes não necessitam apenas de valor acrescentado imediato, precisam também de uma elevada garantia de qualidade (Boehm, 2002). Nos métodos tradicionais tem-se demasiada rigidez nos processos, o que sufoca a criatividade e retarda a adaptação das equipas. Porém, se as metodologias Agile implicarem demasiada flexibilidade podem trazer ineficiência, levando à diminuição da qualidade dos produtos e consequentemente à baixa do seu valor. Assim, um dos principais objetivos da gestão ágil reside na busca permanente do equilíbrio entre a flexibilidade (inerente à agilidade) e a estabilidade (Cobb, 2011). Neste contexto, ao gestor de projeto é atribuído papel primordial enquanto garante da conjugação mais satisfatória entre os dois extremos.

A gestão Agile implica, por isso, uma nova forma de liderança, direcionada ao serviço à equipa que deve ser apoiada com vista a atingir resultados atendendo às necessidades de todos os seus membros. A primeira função de um líder Agile consiste, assim, em remover impedimentos e obstáculos que estejam a impedir a conquista dos objetivos por parte dos elementos da equipa (Stellman & Greene, 2015).

Segundo Hoda *et al.* (2011) a liderança nas equipas auto-organizadas, como as do Agile, tem por propósito ser de orientação subtil e adaptativa, concedendo feedback e direção ténue. No Scrum, por exemplo, o gestor que toma o nome de *ScrumMaster*, é responsável por gerir o processo e proteger a equipa de interferências externas (Singh, 2008). Consequentemente, os líderes das equipas Agile são responsáveis por alinhar as pessoas, obter os recursos necessários e motivar as equipas.

Em sentido lato, a gestão e as metodologias Agile são vagamente detalhadas pois estão, por definição, destinadas a serem adaptadas e customizadas a cada situação e projeto. Assim, cabe ao Agile dar os princípios e valores (Cobb, 2011) e, com base nestes, as equipas verão emergir os processos, as estruturas de trabalho e os princípios aplicados à sua própria realidade (Boehm, 2005).

⁴ Augustine, S. (2005). *Managing Agile Projects*. Prentice Hall, Upper Saddle River, NJ.

Técnicas de Gestão Agile

Todas as técnicas utilizadas na metodologia Agile, também conhecida comumente como uma “metodologia leve”, podem ser aplicadas, igualmente, à gestão tradicional de projeto, permitindo-lhe aumentar eficiência e performance.

Entre as técnicas Agile destacam-se:

A imagem da gestão Agile está associada aos quadros de **controle visual** (ou *whiteboards*) com vários cartões de tarefas coloridos. Trata-se de uma ferramenta utilizada recorrentemente pois permite a auto-organização das equipas e atingir um estado de entendimento comum sobre as tarefas em mãos, possibilitando uma visão comum sobre o do estado do projeto (Cobb 2011).

Outra prática muito importante consiste em dispor todos os elementos chave de uma equipa na **mesma localização**, idealmente na mesma sala ou divisão, incluindo o representante do cliente (que no Scrum tem o nome do *product owner*). Tal procedimento visa a melhoria na comunicação entre os vários elementos e o incremento da qualidade de coordenação. Aqui o gestor de projeto tem a responsabilidade de ajudar a manter um ambiente colaborativo entre todas as partes, isto é, programadores e cliente a trabalhar em torno do objetivo comum. (Boehm 2005 e Cockburn 2004).

Quando o cliente revela dificuldade em articular e expressar os requisitos, as equipas Agile usam a técnica de **desenvolvimento orientado aos testes**. Os casos de testes são desenhados em primeiro lugar e só depois se mapeia contra os requisitos. Isto significa que as quatro fases são colapsadas numa só (requisitos, desenho, desenvolvimento e teste) mas requerem mais iterações (Schwaber e Beedle 2002).

Um ambiente Agile é um ambiente dinâmico e por isso as equipas têm de estar em constante **adaptação** (Angioni et al. 2006), o que pode causar incómodo para pessoas habituados a estruturas rígidas. Aqui, o papel do gestor de projeto revela-se determinante, devendo ser visto como um líder e não como um gestor de tarefas. Terá de trabalhar de modo a que a equipa crie e construa fundamentos para uma boa base colaborativa. A adaptação da equipa é revelante mas mais importante é a reunião das **lições aprendidas** nas ciclos precedentes e que ajudarão a melhorar o seu desempenho nas iterações seguintes (Cobb 2011 e Boehm 2005).

A **colaboração** representa uma técnica e uma palavra-chave quando se fala de Agile. Por um lado, a colaboração da liderança que tem como principal objetivo remover barreiras que dificultem o núcleo da equipa, por outro lado, a colaboração dos próprios membros entre si e com o cliente. A colaboração nesta perspetiva é fundamental para que a equipa entregue resultados, possa captar feedback do que já passou e possa aplicar conhecimento para melhorar no futuro. O planeamento constitui responsabilidade da equipa como um todo e os requisitos são priorizados por todos com o suporte do cliente num ambiente de grande colaboração (Cobb, 2011).

Partir a solução em pequenos incrementos e focar nas **funcionalidades baseando-se no valor** e na receita que estas poderão trazer são boas técnicas ágeis. Ao partir-se a solução complexa em pequenas partes vai-se diminuir a complexidade do problema, favorecendo-se a focagem da equipa **numa funcionalidade de cada vez** enquanto as outras são mantidas em *backlog* (Schwaber & Beedle, 2003). A priorização é feita com base no valor e aumento de receita e quota de mercado. É da responsabilidade da parte do cliente a priorização para que a equipa de desenvolvimento não invista demasiado tempo em funcionalidades que possam não ter relevância (Cobb 2011 e Boehm, 2005).

As **reuniões matinais** de projeto, no SCRUM, *Daily Stand-up Meetings*, são também técnica Agile e têm por principal objetivo garantir que a equipa está alinhada. Essas reuniões, feitas de pé e sem demorar mais do que 15 minutos, pretendem ser bastante focadas e seguir uma abordagem acelerada, sem deixar de garantir que os membros das equipas estão comprometidos e focados nos resultados. Apresentam uma grande vantagem pois promovem a consolidação da equipa e potenciam a comunicação (Sutherland e Schwaber, 2007).

A metodologia Agile assenta fortemente no **consenso**, pelo que é necessário muita facilitação entre os membros da equipa de forma a concordarem nos diversos tópicos. A premissa da concórdia assenta na ideia que o acordo entre todos traz um comprometimento real e a responsabilização dos membros pelas decisões acordadas e ações consequentes. Uma das técnicas aplicadas é a abordagem dos “cinco dedos” em que os membros mostram o seu grau de consenso em redor de um determinado tópico utilizando os dedos da mão e enquanto não houver consenso não há decisão. Se o consenso não for atingido o facilitador terá de organizar outras sessões de forma a se atingir o consenso total em relação ao tópico em discussão (Cobb, 2011).

O agile defende que a aplicação da definição de **blocos de tempo** de forma a fixar a data final para a execução de uma iteração e não permitir que esta se altere, é uma mais-valia (Boehm 2005). Esta técnica, que toma o nome de *timeboxing*, proporciona uma das suas principais vantagens, segundo Cockbun (2000), a de trazer tempo e paz de espírito à equipa de maneira a que ela se dedique em exclusivo e em tornar funcional o *software* que tem em mãos. A dimensão dos blocos de tempo depende da velocidade da equipa e não são necessariamente iguais embora seja isto o aconselhado (Cobb, 2011). Ora, fixando o tempo de iteração a equipa decide quantas funcionalidades podem ser entregues no intervalo de tempo estipulado. Se uma funcionalidade for incluída não poderá mudar durante o seu desenvolvimento onde se faz uma análise mais profunda e detalhada do seu conteúdo. Defensores do *timeboxing* sustentam que a sua aplicação faz com as pessoas ganhem atenção ao trabalho e não ao tempo, aumentando assim a sua produtividade pois a definição de um intervalo de tempo para uma tarefa permite que as pessoas se esforcem mais por trabalhar de forma inteligente (Boehm, 2005).

Pontos Fortes

De todos os pontos já referidos anteriormente quando se abordou a vantagem da aplicação do Agile destaca-se os que implicam maior impacto: capacidade de adaptação a ambientes de rápida evolução, maior qualidade em geral dos entregáveis e superior benefício do cliente.

Pontos Fracos

Segundo Balaji & Murugaiyan (2012), a utilização do Agile revela-se uma abordagem lucrativa quando aplicada a pequenos projetos, não sendo, todavia, escalável para projetos de grande dimensão, uma vez que endereça problemas de planeamento e gestão de alto nível que sempre surgem com projetos com múltiplas equipas.

Um outro ponto fraco apontado reside na elevada dependência das capacidades dos membros da equipa para seguirem os princípios Agile, e isto só funcionará se estes tiverem experiência e conhecimento suficientes para assumir a responsabilidade, o que só se revela possível quando atingem alguma senioridade (Balaji & Murugaiyan, 2012).

Como afirma Constantine⁵ “Existem tantos Kent Beck no mundo para liderar equipas... As metodologias ágeis apostam tudo em pessoas *premium*” (Boehm, 2002).

Kent Beck, criador da metodologia XP foi um dos 17 signatários do Manifesto Agile.

A ausência da função de analista de negócio pode ser considerado, igualmente, um ponto fraco, pois os programadores não estão habituados a seguir o ponto de vista do negócio, fazer a análise de processos que deve ser feita antes ou paralelamente ao desenvolvimento. Em situações complexas até pode chegar a ser necessário uma análise adicional aos requisitos e garantir que estão completos e definidos de forma rigorosa, consistentes com os objetivos de negócio (Cobb, 2011).

Um projeto tipicamente Agile dependente fortemente da comunicação individual frente-a-frente e da memória dos membros da equipa. A ausência de foco na documentação e o no seu controlo (Sutherland & Ahmad, 2011), em certas ocasiões poderá trazer mais custos e defeitos pois a sustentabilidade do sistema fica comprometida pois os requisitos bem detalhados são essenciais à manutenção e gestão do ciclo de vida operacional do sistema (Uikey, Suman & Ramani, 2011).

Além disso, a ausência de documentação pode ser preocupante no caso de algum dos membros da equipa sair não fazendo qualquer passagem de conhecimento ou mesmo, se um novo membro se juntar à equipa, este terá de interromper o trabalho dos restantes de forma a pôr-se a par do trabalho que está a ser realizado, e isso desacelera a equipa Agile. (Uikey *et al.*, 2011).

Devido à agilidade associada a esta metodologia, as equipas são encorajadas a passar rapidamente para execução com pouco planeamento efetuado, isto pode levar a um conjunto de problemas, entre os quais se destaca a elaboração de estimativas de custos pouco rigorosas que podem acarretar surpresas indesejáveis para o projeto.

A abordagem incremental também traz um senão pois implica que a arquitetura poderá não estar completa desde início e poderá evoluir ao longo do processo. Como resultado poderá ser necessário excessivo retrabalho (Cobb, 2011).

A abordagem de gestão que está compreendida nas metodologias ágeis como o SCRUM trata de questões ao nível da iteração e do seu planeamento (ou seja, questões

⁵ Constantine, L. (2001). *Methodological Agility*. Software Development, pp.67-69

mais operacionais). Muitas vezes é necessário um modelo de gestão de projeto acima para englobar as práticas de mais baixo nível, as várias camadas de iteração para que se possa entregar de forma mais robusta soluções grandes e complexas, particularmente aquelas que envolvem a intervenção de múltiplas equipas ágeis. (Cobb, 2011).

Como referido anteriormente muitos autores consideram que as metodologias ágeis não são escaláveis (Balaji & Murugaiyan, 2012). Há porém quem defenda o oposto como é o caso de Sutherland (2001), que, em 1996, quando contratado pela IDX Systems, uma empresa norte americana de *software* de saúde com centenas de programadores a trabalhar em dezenas de equipas com dezenas de produtos, para o cargo de vice- presidente de engenharia e de desenvolvimento de produto, decidiu aplicar o *Scrum-of-Scrums*, que consiste num mecanismo de coordenação básica para planeamento geral de projetos, gestão de equipas que por vezes envolve a negociação de fronteiras de responsabilidade e consenso no que diz respeito às interfaces entre equipas. Tem como objetivo facilitar a resolução de conflitos e elencar dependências entre equipas (Cobb, 2011; Sutherland, (2001) pressupõe uma reunião semanal de cada representante da equipa/produto, enquanto as reuniões de gestão de Scrum são feitas mensalmente. Como resultado desta aplicação, Sutherland (2001) concluiu que a maioria das equipas apresentava uma produtividade considerada média para a indústria, mas existiam algumas equipas que passaram para um estado de hiper-produtividade, produzindo entregáveis 4 ou 5 vezes mais rapidamente que a média. A maior dificuldade, segundo ele, foi o processo da implementação da metodologia SCRUM na IDX e atingir a qualidade nas práticas ao nível da equipa, não ao nível do *Scrum-of-Scrums*. O aspeto mais importante desta implementação foi concluir que de facto o Scrum é uma metodologia Agile pode também ser escalável (Sutherland, 2001).

1.4 Comparação Gestão Tradicional e Agile

Tanto as metodologias tradicionais como as Agile, frequentemente comparadas em discussões destinadas a decidir sobre quais as práticas mais adequadas a cada organização ou projeto, serão abordadas a seguir, procurando-se definir as suas principais diretrizes e proceder a uma resumida apreciação comparativa.

Em traços muito gerais poderá começar por afirmar-se que enquanto as metodologias tradicionais arrancaram com a atribuição de uma orientação muito forte ao processo, incorporando, depois, o princípio da adaptação do processo às pessoas, se bem que consentido que elas permanecessem em segundo plano (Cobb, 2011), as práticas Agile, privilegiando uma intenção humanística de orientação às pessoas, começaram por dar realce aos indivíduos, cujos direitos e interações ganham o estatuto de sobreposição aos processos e ferramentas (Beck, 2001).

Seguidamente apresenta-se uma tabela de sumarização das principais diferenças entre a gestão tradicional de projetos e a Agile (Ziólkowski & Deregowski, 2014; Lyton, 2012; Balaji & Murugayan, 2012; Sutherland & Ahmad, 2011; Sutherland, 2001; Highsmith, 2002):

Tradicional	Agile
A garantia de sucesso depende da entrega do projeto dentro do âmbito.	A garantia de sucesso consiste na adaptação à constante mudança de necessidade por parte do cliente.
Abordagem prescritiva, que tenta prever o futuro, promovendo um plano baseado em premissas destituídas de dados empíricos.	Abordagem empírica baseada na teoria de controlo do processo. Gestor (<i>ScrumMaster</i>) e equipa procuram adaptar-se aos resultados, ajustando as iterações aos objetivos de entrega e à velocidade da equipa.
Planeamento <i>button-up</i> - garante ênfase às tarefas individuais e ao plano no início de cada projeto.	Abordagem <i>Top-down</i> - durante o desenvolvimento descobre e elabora progressivamente um plano.

Produção de muitos artefactos documentação pesada e esmagadora.	Documentação pragmática, simples minimalista, organizada e otimizada para aumentar a probabilidade de ser lida por alguém.
Estrutura organizacional corporativa formal com muitas regras para cumprir.	Estrutura plana cuja autoridade depende da experiência e capacidades técnicas, em vez de títulos corporativos
Progresso do projeto medido pelo gestor.	As equipas auto-organizadas utilizam ferramentas para darem o feedback acerca do progresso das tarefas que têm em mãos.
Gestão dos <i>Stakeholders</i> e planeamento das suas intervenções.	Envolvimento dos stakeholders. A gestão divide responsabilidades do gestor de projeto com a equipa e com o gestor de produto.

Da deficiente interpretação de orientações ou de uma pobre implementação das metodologias em apreciação resulta o estabelecimento de alguns estereótipos no seio da indústria de *software*. A gestão tradicional é, por exemplo, tida como portadora de muitos processos incómodos que gerem as pessoas, em vez das pessoas gerirem os processos (Hass, 2007). Por outro lado, o Agile está associado à crença negativa de se tratar de uma metodologia que não tem qualquer processo, ou que é caótica, ficando facilmente fora do controlo (Highsmith, 2002). Alguns críticos desta metodologia chegam mesmo a sustentar que são pouco profissionais, não dispendo de substância e não funcionando para projetos complexos por ser destituída de práticas de gestão de alto nível (Boehm & Turner, 2003).

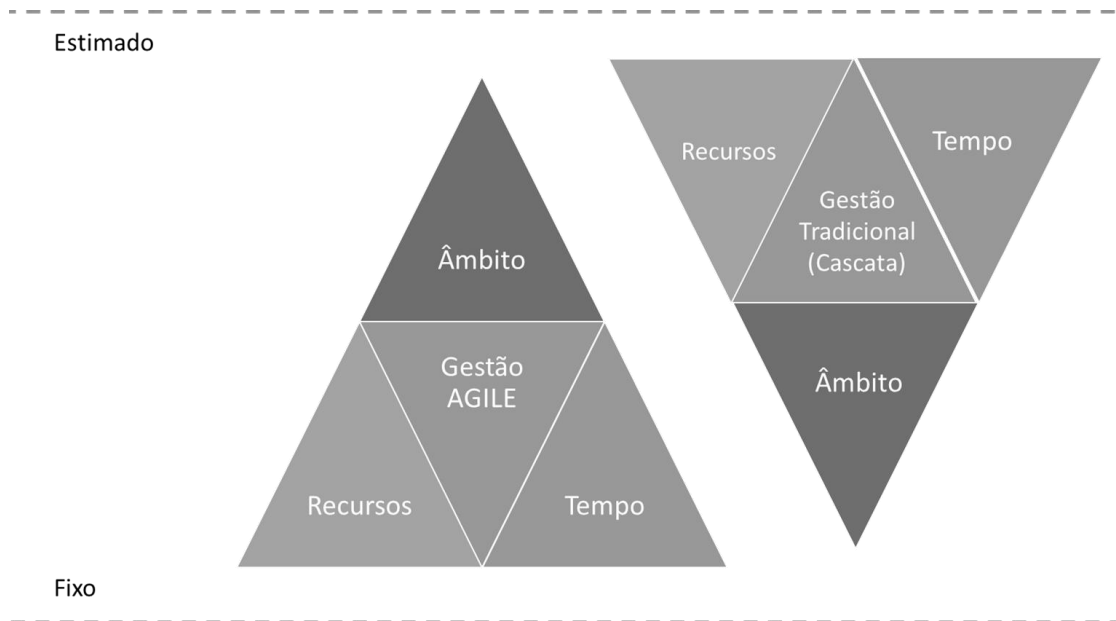


Figura 4: Adotação do triângulo de ferro tradicional, publicado originalmente por Martin Barnes em 1969 (PMI, 2013).

Se muitas das falhas atribuídas às práticas tradicionais de Agile resultarão da sua má aplicação, outras serão fruto das suas próprias limitações. Metodologias tradicionalistas orientadas ao plano perdem prestígio quando exageram na documentação e nos controlos burocráticos em vez de tomarem uma abordagem inteligente de adaptar o nível de documentação e os controlos do projeto e ambiente de negócio de forma a reduzir o risco. (Cobb, 2011). O Agile, por seu lado, perde consideração quando as equipas avançam precocemente para a execução de projetos sem um mínimo de planeamento, não havendo nada na metodologia que estabeleça a quantidade de planeamento necessário antes de começar a execução (Boehm, 2002).

Outro ponto frágil atribuído ao Agile é o potencial conflito resultante para as organizações tipicamente tradicionais com CMMI nível 5 pois, apesar destas já seguirem o conceito de constante adaptação e mudança, a documentação exigida pela adoção do Agile nem para o primeiro nível é considerada adequada (Boehm & Turner, 2005). Ora, tal circunstância pode gerar preocupação e incómodo para essas organizações conseguirem manter a avaliação ou *rate* CMMI e, ao mesmo tempo, se tornarem mais ágeis.

As metodologias Agile têm vindo a amadurecer ao longo tempo e a reconhecer que se torna necessário adotar um processo de disciplina em adição à forte orientação ao

valor e às pessoas (Highsmith, 2002; Boehm, 2002), procurar uma abordagem mais equilibrada que misture a quantidade certa de controlo em relação ao plano, agilidade e adaptabilidade aos novos requisitos de negócio, à medida que o projeto progride (Cobb, 2011). Apesar da visível progressão que as metodologias ágeis têm vindo a registar, coexistem certos extremos que defendem que o planeamento e os custos não são importantes, como refere Ron Jeffries, um grande contribuinte do XP “Os grandes *designs* emergem de nenhuma antecipação ou planeamento e de re-factorização contínua” (Highsmith, 2002).

Muitas organizações consideram as metodologias tradicionais e ágeis como mutualmente exclusivas. Porém, analisadas de perto, constata-se uma coincidência de objetivos, especificamente em aspetos como a grande aposta na qualidade, maior satisfação do cliente e redução o *time-to-market* (Cobb, 2011). Se se considerar que, devido às características de complexidade e multidimensionalidade cada projeto de IT é único, deve atender-se à conveniência da adoção de uma abordagem com regras, práticas e procedimentos adaptados de ambas as metodologias com vista a se obter sinergias e um processo flexível que consiga selecionar os problemas e desafios do dia-a-dia. (Boehm, 2002; Cobb, 2011; Highsmith, 2002).

1.5 Lean

1.5.1 Nascimento do Lean na Toyota

Tem quase cem anos de aplicação um dos conceitos fundamentais do sistema Lean - resumidamente designado de «*Jidoka*», Automação Inteligente, sendo implementado inicialmente pelo fundador do grupo Toyota, Sakichi Toyoda, em 1937. Produção *Just-in-time* é outra das ideias básicas do Lean, que do construtor automóvel japonês se estendeu à maioria das indústrias do mundo, alargando-se as suas práticas a atividade e setores tão distintos como departamentos de defesa de Estado, hospitais, instituições financeiras e empresas de construção (Liker, 2006).

Em 1941 a II Guerra Mundial viria a interromper os esforços de Sakichi, que consistiam em reformular os métodos americanos de produção em série e, terminado o conflito, de que o Japão saiu quase completamente destruído, a Toyota enfrenta uma grande crise de vendas cuja responsabilidade seria assumida por Sakichi, que decidiu passar o seu projeto ao primo Eiji. (Womack, Jones & Roos, 2007). A empresa redefiniu então os seus objetivos no sentido de tornar a fábrica mais próxima dos padrões americanos, o que incluía um alinhamento com os *standards* tecnológicos em apenas três anos.

Uma tal orientação determinou que, em 1951, Eiji se deslocasse aos Estados Unidos, para aprender as práticas mais avançadas no sector automóvel, nas unidades de produção da Ford em que passou três meses (Womack et al. 2007). Na altura, a produtividade e a produção de automóveis nos EUA eram oito vezes superiores às do Japão, além de que, a Toyota não disponha nem de capital nem equipamento ao nível dos norte-americanos.

É neste contexto, que Taaichi Ohno, gestor de oficina Toyota, se vê encarregado de desenvolver um sistema de produção mais eficiente. Durante duas décadas os engenheiros Eiji e Taaichi irão empenhar-se na aplicação do *Jidoka* em todas as tarefas operacionais do construtor nipónico, ao mesmo tempo que procuraram encontrar formas alternativas de aumento dos recursos limitados e de capital, aplicando sistematicamente o conceito *just-in-time*. Desses esforços resultou um sistema unificado

de alta produtividade e qualidade superior o chamado Sistema de Produção Toyota (SPT).

O SPT tornar-se-ia um *standard* para sistemas de produção, ganhando, depois da crise do petróleo de 1973, especial atenção internacional, visto que a Toyota surgiu entre as indústrias automóveis que mais rapidamente recuperou da crise. Muitos especialistas do setor atribuíram um tal desempenho ao SPT.

Em consequência desse êxito, vários consultores de produtividade e empresários começaram a introduzir o sistema nos Estados Unidos. Em 1990 James Wormack, consultor de produtividade utilizaria pela primeira vez, na sua obra *A Máquina que mudou o mundo*, o termo Lean para designar o sistema criado e aperfeiçoado ao longo de décadas pela Toyota. Por esse nome ficaria mundialmente conhecido o Lean, um sistema de produção e gestão considerado eficaz especialmente pela capacidade de gerar melhorias significativas em termos de produtividade e de qualidade e que não revela sinais de desaceleração (Liker, 2006).

Significado

Lean significa priorização e flexibilidade de processos, eliminação de excessos de produção, corte nas gorduras e tempos de espera, rapidez de resposta aos pedidos (Castro, 2012).

Por utilizar recursos, qualquer atividade de uma organização repercute-se na eficiência e produtividade do respetivo uso. Importa, por isso, assegurar uma alocação de recursos a atividades que realmente tragam valor de forma a reduzir o desperdício. Consequentemente, o *Lean* defende a ideia de que o valor deve ser definido pelo cliente para que se possa distinguir claramente se uma atividade traz ou não valor acrescentado ou se deve ser eliminada (Poppendieck & Cusumano, 2012).

Assim a meta final das organizações *Lean* consiste em tornarem-se fornecedoras preferenciais, empregadoras preferenciais e, mais importante, pontos de investimento preferenciais.

Para seguir um dos princípios fundamentais do *lean*, Especificar o Valor, e ***eliminação do desperdício***, deve começar-se por identificar o valor, o que traz valor acrescentado e paralelamente ir identificando o que se considera desperdício.

Segundo as regras do sistema, uma atividade tem valor acrescentado quando, cumulativamente, obedecer aos três critérios seguintes (Castro, 2012):

- O cliente está disposto a pagar pela atividade;
- Surgir associada a uma transformação “física” do produto;
- For bem executada à primeira, garantindo zero defeitos.

Produzir algo que o cliente não pediu, o que não é necessário ou fora da época adequada consideram-se desperdício e trata-se de *excesso de produção*. De notar que produzir algo quando não é altura certa para tal corresponde a ocupar/utilizar recursos com algo que não é ainda importante, existindo a possibilidade de o não vir a ser nunca.

Todo o produto ou serviço à *espera* de algo com impacto no ciclo da organização é considerado um desperdício. Por exemplo, a espera por aprovação de um documento para tomar uma decisão (Jugulum & Samuel, 2008). Além de ser um desperdício, o facto de os colaboradores estarem à espera para executarem o seu trabalho tira-lhes a sensação de bem-estar e motivação.

O *transporte* de bens ou pessoas é considerado um desperdício quando é desnecessário, não adiciona valor, deve ser eliminado ou reduzido (Jugulum & Samuel, 2008).

A *criatividade não utilizada* trata-se de um tipo de desperdício muito difícil de medir e, conseqüentemente, de avaliação complexa quanto ao seu impacto. Refere-se a todas as competências de uma organização em que não são utilizadas e/ou ideias que nunca chegam a ser ouvidas. Entre as causas possíveis deste desperdício, estão tipos de liderança, cultura da empresa e forma de comunicação, entre outras (Castro, 2012).

Quando se deteta um *defeito* num produto ou serviço este deve ser corrigido ou enviado para o lixo. O trabalho e o custo associado apenas ao retrabalho poderá muitas vezes exceder o valor de executá-lo corretamente à primeira.

Um conceito importante sobre a perspectiva *lean* reside no que se considera custo de um serviço ou produto. Numa visão tradicional o custo corresponde à soma dos custos de todas as atividades envolvidas na produção do serviço ou produto, mas segundo *lean* não, apenas se devendo considerar os encargos das atividades de valor acrescentado (Castro, 2012). Tudo o que está associado a custos provenientes do desperdício devem ser eliminados porque se sabe que o cliente não pretende pagar por eles.

Outro princípio Lean consiste na *identificação da cadeia de valor* que consiste no conjunto de processos e etapas que um produto, serviço ou informação tem de percorrer para chegar ao seu destino final. Após a identificação da cadeia de valor, torna-se necessário medir o seu desempenho, pois tanto as atividades de valor como o desperdício existem lado a lado. A medição do desempenho deve ser feita à luz dos critérios da organização e do que é realmente importante para o cliente.

Revela-se muito importante identificar aquilo que acontece no processo e o que se vê numa cadeia de valor, bem como identificar e especificar processos e etapas que não são bem visíveis.

A abordagem Lean defende a criação de um *fluxo contínuo* e para isto é necessário ter um pensamento global da organização e deixar de pensar individualmente por departamentos. Não se afigura correto pensar que a maximização do sistema global se consegue com a maximização individual ao nível dos departamentos, pois tal só seria verdadeiro se os departamentos fossem completamente independentes entre si.

Segundo a teoria das restrições (teoria incorporada no *lean*) uma corrente é tanto mais forte quanto mais forte for o seu elo mais fraco, visto que se este se parte a corrente deixa de funcionar (Castro, 2012).

Pensando em processos sequencias executados por equipas distintas, com os quais se pretende melhorar o processo global, a aplicação de esforços de melhoria em processos que não são os piores é desperdício pois qualquer melhoria não terá impacto ao nível global. O foco deve ser o elo mais fraco, pois é ele que dita a velocidade a que funciona todo o sistema.

O *sistema puxado* (pull) é uma ferramenta Lean de especial importância, que apenas produz quando existe uma ordem para tal. Se essa ordem não surgir então o processo não deve produzir, posto que é o cliente que faz o sistema avançar quando consome o produto. Associadamente, tem-se também a ideia do JIT (*Just-in-time*), para além do cliente puxar o produto/serviço este deve ser entregue aquando do pedido, ou seja a produção e entrega serão sempre consequência de demanda e não ao contrário (Castro, 2012).

Aplicação sistemática dos quatro princípios anteriormente definidos numa perspetiva de *melhoria contínua* leva ao aperfeiçoamento do sistema, sendo esta a última orientação Lean, estabelecida com vista a promover a busca pela perfeição (Jugulum & Samuel, 2008). Melhorar implica reestruturar os serviços/produtos para que os defeitos não aconteçam, isto é, construir sistemas à prova de erro (*poka-yoke*)

(Castro, 2012). A abordagem Kaizen (que em Japonês significa melhoria contínua e incremental) é uma ferramenta de implementação deste princípio, consistindo na reunião de um pequeno grupo de trabalhadores para executar o mapeamento do estado atual dos processos de que resulta a criação de uma base, a que se segue o estabelecimento da visão do estado futuro (melhoria). Paralelamente identificam as lacunas entre os dois estados e trabalha-se na implementação das alterações, para que, de forma sistemática se atinjam melhorias contínuas (Jugulum & Samuel, 2008).

Em síntese, nascido no seio da indústria automóvel e nos últimos anos vastamente aplicado a todo o tipo de negócio, como logística, serviços e tecnologia, o *Lean* procura garantir um processo composto de 100% de atividades que tragam valor acrescentado (Janes & Succi, 2014) e produzir apenas aquilo que o cliente procura, *just-in-time*, a um preço competitivo com zero defeitos. Esta abordagem entende que só assim a empresa pode caminhar iterativamente para a sustentabilidade.

1.5.2 Lean e as Tecnologias de Informação

Qualidade da informação e a eficácia dos sistemas de informação revelam-se imprescindíveis para o sucesso da empresa moderna, embora se tenha vindo a verificar um desalinhamento crónico das atividades de IT com estratégia de negócio, o que tem como consequência gastos avultados (Bell & Orzen, 2011).

Segundo Bell & Orzen (2011) o desalinhamento do negócio e o IT são causados pela falta de integração e sincronização resultantes da complexidade desnecessária dos processos. A presença de práticas rigorosas e labirínticas em cada uma das fases e departamentos estão a tornar todo o processo num “Colete Salva-vidas de cimento” (Kumar, 2005).

Os processos foram-se complicando em resposta ao aumento da complexidade do mundo nos seus novos desafios, como a internet, mercado digital e competitividade em aumento progressivo e contínuo. Lidar com o novo contexto, a aplicação dos princípios de gestão *Lean* aposta em processos focados no valor que vão entregar aos clientes e na busca contínua de melhoria de eficiência e eficácia (Janes & Succi, 2014).

Lean IT ou Agile?

O Lean e o Agile partilham os mesmos princípios: aumentar a produtividade no desenvolvimento de *software* enquanto simultaneamente se aumenta a qualidade do *software* resultante. As práticas que fazem parte do Agile também suportam os princípios Lean, mas o Lean tem uma visão mais ampla, preferindo olhar para todo o contexto de negócio no qual o *software* é desenvolvido, enquanto o Agile se foca no processo de desenvolvimento (Hibbs et al., 2009).

Tanto o desenvolvimento Lean como Agile têm como objetivo melhorar a qualidade do *software*, como este é percebido pelo cliente, bem como a produtividade do processo de desenvolvimento. Valorizam, e aceitam bem, as alterações aos requisitos que irão ocorrer de forma mais certa durante o decorrer do projeto. Ambas colocam o maior valor na entrega de *software* que vai ao encontro das necessidades do cliente (não as necessidades do cliente que foram apercebidas inicialmente) (Hibbs et al., 2009).

O Lean IT representa uma forma de pensar (Hibbs et al., 2009; Poppendieck, M. & Poppendieck, T., 2013; Stellman & Greene, 2015) e os seus princípios aplicam-se a qualquer âmbito, desde da prática específica de desenvolvimento de *software* até à organização completa onde o desenvolvimento de *software* é apenas uma parte. Um aspeto chave desta forma de pensar reside no facto de que quando mais amplo for o âmbito da aplicação Lean numa empresa mais amplos serão os potenciais benefícios (Hibbs et al., 2009).

A colaboração próxima com o cliente e na entrega rápida de *software* funcional assim que possível é a principal preocupação do Agile, o Lean considera isso valioso mas o seu foco primário consiste na eliminação do desperdício no contexto daquilo que o cliente valoriza, como se verá em maior detalhe, mais adiante. (Hibbs et al., 2009).

O Agile revela um bom conjunto de metodologias formais, o Lean não dispõe de metodologia formal, revelando, em vez disso, um conjunto opcional de ferramentas de boas práticas recomendadas (Hibbs et al., 2009).

Muitos gestores demonstram ainda algum receio de passar a ter um processo de desenvolvimento Lean ou Agile, considerando não valer a pena arriscar a sua carreira em nome de algo mais ágil. Sustentam que o modelo em cascata garante, em contrapartida, uma abordagem bastante cautelosa.

No entanto, as organizações que já deram o “salto” tornaram-se mais eficientes no desenvolvimento de produtos e no desenho de produtos mais valorizados. Não sendo necessariamente as organizações que entregam mais funcionalidades as que garantem

entregas adequadas (Poppendieck, M. & Poppendieck, T., 2013), a eficiência do desenvolvimento do produto não estará na economia de custos, contrariamente aos que sustentam a ideia e que associam o Lean a baixos salários, despedimentos, e desenvolvimento em Outsourcing.

O foco da eficiência estará, antes, nas mentes criativas e no trabalho de desenvolver produtos bem-sucedidos (Hibbs et al., 2009), existindo uma diferença importante entre a eficiência dos recursos e a eficiência do fluxo de trabalho: a primeira mede-se pelo grau de ocupação dos recursos humanos, que, quando se tenta otimizar, podem obter-se efeitos adversos como o aumento das filas e o desperdício; a segunda, adotada pelas organizações realmente eficientes, assegura uma visão holística do sistema traduzindo-se, inevitavelmente, em melhores resultados de forma geral. Os líderes *Lean* entendem, por isso, que a eficiência provém do fluxo, da velocidade e da aprendizagem de colaboradores que se preocupam (Poppendieck, M. & Poppendieck, T., 2013) e são respeitados. Será com tais ativos que se constituem as equipas de alta performance Lean IT, dotadas de liberdade para decidir e capacitadas para efetuar o trabalho contido dentro do âmbito do projeto com a mínima supervisão e interferência (Martin, 2009). Essas equipas assumem-se como base sustentável para a criação e entrega de sistemas Lean IT.

São várias as soluções de tradução dos princípios Lean originados na indústria automóvel em práticas de engenharia de *software* (Janes & Succi, 2014), destacando-se, entre elas as que são abordadas nos trabalhos pioneiros de Mary e Tom Poppendieck (2003) e, posteriormente, de Curt Hibbs *et al.* (2009) e Stellman & Greene (2015), abordados a seguir.

1.5.3 Princípios Lean em IT

1. *Eliminação do desperdício*

Nem sempre é fácil ver, reconhecer desperdício ou mesmo admitir que uma equipa gasta horas ou dias produzindo algo que não acrescenta praticamente nenhum valor ao projeto. Além disso há que considerar que avaliação e a deteção do desperdício transvasa o próprio projeto (Stellman & Greene, 2015). Como visto anteriormente no exemplo industrial, o Lean tem como maior impulso a eliminação do desperdício (Hibbs et al., 2009; Martin, 2009), preocupando-se, por isso, com a eliminação do que não

acrescenta valor dentro e fora da equipa, tendo em conta os objetivos do projeto (Poppendieck & Cusumano, 2012).

Os desperdícios de produção na área do desenvolvimento de *software* foram mapeados, pela primeira vez, por Mary e Tom Poppendieck em 2003, ficando assim listados:

1.1. Trabalho parcialmente concluído

Na opinião de Stelman & Greene (2015) qualquer atividade que não entregue valor, que não esteja 100% completa e a funcionar representa desperdício, ou, por outras palavras, todo o *software* parcialmente completo tende a tornar-se obsoleto, visto que não se tem consciência se algum dia poderá vir a funcionar quando estiver finalizado ou mesmo resolver algum problema de negócio (Poppendieck, M. & Poppendieck, T., 2003). O trabalho parcialmente concluído apenas serve para despender recursos e o Lean sugere a escolha de uma funcionalidade de cada vez e a utilização do fluxo contínuo de uma única peça. Por esta via garante-se que a funcionalidade segue até o fim do ciclo só sendo dada por concluída quando estiver pronta para ir para produção, isto é, desenvolvida, documentada, testada e livre de erros (Hibbs et al., 2009).

1.2. Processos Extra

Para Hibbs et al. (2009) os processos desnecessários constituem puro desperdício, não acrescentam valor e impedem a produtividade de florescer. Entre os processos deste tipo surgem tarefas manuais que poderiam ser automatizadas, procedimentos complexos para executar tarefas simples, produção de documentos que ninguém lê e procedimentos que não levam a lado nenhum. O excesso de burocracia e a documentação consomem recursos, retardam o tempo de resposta e podem esconder alguns problemas de qualidade (Poppendieck, M. & Poppendieck, T., 2003). A boa documentação, segundo estes autores, pode identificar-se quando há alguém à espera de que esta seja produzida para que se possa avançar com o seu trabalho. A produção de documentos detalhados com qualidade poderá ser muito morosa, existindo, porém, alternativas, como o é a escrita de requisitos em forma de testes de aceitação com o cliente.

Outro exemplo de desperdício na gestão revela-se quando se gasta muito tempo e esforço na construção de um grande gráfico de Gantt ou noutro tipo de plano de

projeto, que nunca traduz a realidade do projeto de forma rigorosa porque é baseado em estimativas e em informação inicial que é alterada significativamente ao longo do tempo e está desatualizada assim que a equipa começa a trabalhar na construção do *software* (Stellman & Greene, 2015). Geralmente o plano de projeto nem é utilizado pela equipa de desenvolvimento, e apesar deste tipo de construções ser necessário, não é útil para a construção do *software* e por isso, do ponto de vista do projeto é considerado desperdício.

1.3. Funcionalidades Adicionais

Os programadores gostam, por vezes, de acrescentar funcionalidades extra ao sistema, umas vezes para se adiantarem ao que julgam que será as necessidades do cliente (Martin, 2009) outras por entusiasmo numa nova tecnologia cuja experiência e capacidades pretendem ter (Stellman & Greene, 2015). Isso parece inócuo, mas corresponde a um verdadeiro desperdício (Poppendieck, M. & Poppendieck, T., 2003). Ao longo do ciclo de vida do *software*, o custo da componente de escrita do código é provavelmente o menor. O código para além de escrito deve ser desenhado, documentado, mantido - alterado, melhorado, corrigido. Segundo Hibbs et al. (2009) a regra dos 80/20 aplica-se à maioria dos produtos de *software*, o que significa que 80% das funcionalidades de um produto nunca são utilizadas e o tempo gasto aplicado a criar este desperdício seria muito melhor gasto aplicando nas verdadeiras necessidades do cliente.

1.4. Troca de tarefas

Trocas de tarefas e interrupções suprimem a produtividade. É necessário algum tempo para que o cérebro se foque numa tarefa, comece a entendê-la e daí que se inicie o processo de resolução do problema. As interrupções provocam o reinício deste processo, e para voltar ao que anteriormente estava a executar é necessário tempo para atingir o mesmo nível de produtividade (Poppendieck, M. & Poppendieck, T., 2003; Hibbs et al., 2009). Os membros das equipas, por vezes, sentem que já têm um emprego a tempo inteiro, como o desenvolvimento de software, e lhes ainda é exigido que deem suporte, formação etc, todo este trabalho com prioridade máxima. Esta troca de tarefas tem um efeito nefasto, requer uma sobrecarga cognitiva o que leva ao desperdício (Stellman & Greene, 2015).

A afetação de muito trabalho a uma organização que desenvolve *software* corresponderá à abertura uma fábrica de produção de resíduos. O trabalho, à semelhança dos fluídos, progride mais rapidamente através de canalizações que não estão cheias até a sua capacidade máxima. (Poppendieck, M. & Poppendieck, T., 2003).

É também por isso que a abordagem Lean defende o fluxo de trabalho de peça única (Hibbs et al., 2009) de modo a começar e prosseguir sempre com a mesma tarefa/funcionalidade até ao fim para eliminar o desperdício das trocas.

1.5. Tempo de espera

Durante um processo de desenvolvimento desperdiça-se muito tempo à espera de algo por acontecer, estando identificadas variadas fontes de atrasos que vão da demora no arranque dos projetos e disponibilização das pessoas, a delongas provocadas pela documentação, excesso de requisitos, revisões e aprovações (Poppendieck, M. & Poppendieck, T., 2003). Estes atrasos provocam perda de tempo no processo e aumento dos custos de produção (Martin, 2009).

Inevitavelmente o programador tem de colocar questões aos outros colegas, aos clientes e a outras partes interessadas pois não é detentor de todo o conhecimento do projeto. Se estas pessoas não estão imediatamente disponíveis irá haver redução da velocidade de desenvolvimento (Hibbs et al., 2009). Nesta situação o programador, de forma a continuar a ser produtivo irá trocar de tarefa (mas as trocas de tarefa são desperdício, como vimos anteriormente) ou então irá tentar supor a resposta para continuar o seu trabalho, correndo o risco de ter de o refazer caso a sua suposição esteja errada. Alternativamente, poderá tentar descobrir a resposta por ele próprio, no entanto, se esta tarefa exigir muito esforço ele terminará a supor a resposta na mesma para evitar demasiados obstáculos. De modo a colmatar esta situação, torna-se essencial trazer as respostas até ele, criando-se uma *Equipa Integrada* em que todos os membros estão juntos, localizados na mesma geografia.

1.6. Movimento

Na perspetiva *lean*, importa saber se o cliente consegue dar vazão ao número de questões colocadas pelas equipas e/ou se sente dificuldades. Os ciclos Agile promovem um espaço de trabalho comum (cliente e fornecedor), procurando a otimização do movimento (Poppendieck, M. & Poppendieck, T., 2003). O processo

clássico em cascata está cheio de *handoffs* ou transferências. Os analistas criam o documento de requisitos que passam aos arquitetos, seguidamente estes criam os requisitos técnicos desenham o produto e passam-nos aos programadores que por sua vez implementam o código e passam-no para os *testers* que o validam contra os requisitos. Uma grande quantidade de conhecimento é perdido neste processo porque é impossível guardar num documento tudo o que se aprendeu, descobriu e se criou (Poppendieck, M. & Poppendieck, T., 2003; Hibbs et al., 2009). E este entendimento incompleto vai levar a erros e omissões que resultarão num aumento do custo derivado ao retrabalho.

Segundo Stellman & Greene (2015) quando a equipa não se envolve e não há comunicação entre os seus elementos de forma a discutir os temas, há um acréscimo de movimento, ou seja, são adicionados dias ou semanas ao projeto se os membros das equipas desperdiçarem tempo divagando.

1.7. Defeitos

O foco do ambiente Lean é a prevenção de defeitos contrapondo-se ao tradicional cujo foco é encontra-los depois de eles já terem ocorrido. Os defeitos podem resultar de má comunicação com o cliente ou mesmo tradução deficiente de especificações (Martin, 2009).

A quantidade de desperdício causado por um defeito tem que ver com o impacto que produz e o tempo em que permaneceu sem ser detetado. Um defeito crítico detetado em minutos não representará um grande desperdício, mas um defeito menor que não é descoberto por semanas poder assumir como um grande desperdício. O esforço da qualidade deve, por isso, concentrar-se na detenção precoce de defeitos, através de pequenas iterações (Poppendieck, M. & Poppendieck, T., 2003) mas quando estes são encontrados deverá ser averiguada a sua origem e efetuadas as alterações para garantir que não voltem a ocorrer (Hibbs et al., 2009; Stellman & Greene, 2015).

Muitas vezes o desperdício existe mas não é tarefa fácil encontra-lo. De forma a facilitar a sua deteção existe um exercício simples, o chamado VSM (*Value Stream Map*) cuja origem provém do Lean industrial e que faz todo o sentido aplicar às equipas de desenvolvimento de *software* (Martin, 2009). Começa-se por escolher uma unidade

de valor pequena que a equipa já tenha construído e entregue aos utilizadores ou clientes. Depois disso há que se pensar em todos os passos que esta pequena unidade passou desde que surgiu até à sua entrega. Seguidamente, estima-se a duração do trabalho em cada passo e quanto tempo decorreu depois de findo um passo até ao início do próximo. O VSM permite ter uma noção clara do tempo envolvido num projeto, incluindo o tempo de execução real da tarefa e os tempos de espera. Este mapa permite avaliar, sem todo o detalhe explicativo do sucedido e as razões dos tempos de espera, quais dos atrasos se devem a desperdício e quais os que foram mesmo necessários para a execução projeto.

2. Construir com Qualidade

Cada passo do processo deve ser construído à prova de erro. Quando um problema era detetado na linha de montagem da Toyota esta parava só sendo retomada quando a causa do problema estivesse resolvida e corrigida, para que não ocorresse novamente (Poppendieck, M. & Poppendieck, T., 2003).

O desenvolvimento tradicional de *software*, deixa os defeitos “escorregar” para dentro do produto e só são apanhados durante as inspeções de qualidade (Hibbs et al., 2009).

A abordagem Lean pretende tornar o código à prova de erro e escrever testes à medida que se programam as funcionalidades, que ajudam a prevenir alterações subsequentes ao código com custo mais elevado. A deteção destes erros precocemente aumenta a produtividade da equipa.

3. Amplificar o conhecimento

As Opções de Pensamento ou *Options Thinking*, segundo Stellman & Greene (2015) constituem a ferramenta Lean que permite deixar em aberto algumas componentes do plano, ao contrário do que deve acontecer com a gestão tradicional cujo compromisso sobre todos os aspetos dos projeto representa uma exigência das fases iniciais. Tal prática assegura liberdade à equipa para decidir como planear o seu trabalho e desenhar o seu software.

O desenvolvimento baseado em conjuntos «*Set-based development*» segundo Stellman & Greene (2015) é outra ferramenta Lean pensada para ajudar a equipa a lidar

com o fato de não se saber *A priori*, antes da experimentação, qual das soluções disponíveis irá melhor resultar e será mais rentável quando aplicada a um determinado problema. Assim, em vez de escolher um único caminho, a equipa persegue duas vertentes até chegar a uma conclusão, o que, podendo parecer inútil à primeira vista, representará eventuais poupanças de esforço para equipa que, por essa via, se revelará mais produtiva logo que ultrapassada a fase inicial do projeto.

Finalmente, um dos pontos relevantes deste princípio consiste em não esquecer as lições aprendidas, apostando-se sempre na procura de formas de registar o conhecimento da equipa de modo a ter-se-lhes fácil acesso assim que necessário (Hibbs et al., 2009). Um exemplo prático está na documentação das decisões a nível de arquitetura bem como o detalhe sobre os fatores que foram considerados nas escolhas aquando o seu desenho. Tal conhecimento que pode, algumas vezes, resultar em poupança de tempo no futuro, noutras, se for detalhado em demasia, pode provocar o efeito contrário, sendo importante manter o equilíbrio e bom senso.

4. Adiar o compromisso

Se as melhores decisões se tomam quando se tem conhecimento da maior quantidade de informação disponível (Hibbs et al., 2009), segundo a abordagem Lean, deve esperar-se até ao último momento considerado responsável para se decidir, revelando-se desaconselhável esperar em demasia, porque outras partes do projeto não podem atrasar sob pena de perda de eficiência (Stellman & Greene, 2015). O tempo que se ganha será bem utilizado se se explorar as características das outras opções, o que à partida aparenta ser desperdício, mas as oportunidades bem aproveitadas podem resultar em grandes ganhos de eficiência.

5. Entregas rápidas

O desenvolvimento de *software* surge muitas vezes associado a um empreendimento abstrato (Hibbs et al., 2009), visto que a maioria das pessoas lida melhor com coisas concretas que conseguem ver e com as quais conseguem interagir. Além disso, quando o trabalho se traduz em algo real torna-se mais fácil avalia-lo e verificar se funciona ou não. Sendo os requisitos de *software* bastante voláteis por

resultam da construção de mentes que imaginam algo no plano teórico mas que, quando se lhes é apresentado em concreto, implica o aparecimento de ideias para novas soluções, e por isso o modelo de produção em cascata releva-se bastante propenso à falha.

Talvez por isso, entrega rápida, traduzida na disponibilização de pequenas quantidades ao cliente, permitirá eliminar uma quantidade tremenda de desperdício e retrabalho criada pela acumulação de requisitos.

Uma vez esgotada a capacidade de acelerar as tarefas de desenvolvimento, o Lean contribui com algumas técnicas e ferramentas, como por exemplo, a ‘teoria das filas’ ‘o custo do atraso’ (Stellman & Greene, 2015) e o ‘sistema puxado’ (Poppendieck, M. & Poppendieck, T., 2003).

Relativamente à ‘teoria das filas’ o Lean sustenta que se deve colocar a fila de trabalho do conhecimento de todos e no centro processo de tomada de decisão para que a equipa se possa ajudar internamente e integrar com maior velocidade. Sendo que um sistema sobrecarregado evidencia pelo menos uma restrição e quando as tarefas se acumulam sobre uma equipa se espera que os seus membros executem múltiplas tarefas ao mesmo tempo, o que resulta em desperdício, o Lean defende o uso do sistema puxado que utiliza as filas e as acumulações para eliminar as restrições. O sistema assume tal designação porque os membros entre equipas ou intra-equipas apenas puxam para si as partes/tarefas de que necessitam para prosseguir com o seu trabalho. Um bom exemplo disso está no facto de que quando os programadores acabam de codificar uma *User Story* podem ‘puxar’ a próxima sem ter de receber-las todas ao mesmo tempo, num documento massivo, que poderá só até chegar quando for tarde demais para começar a desenvolver. Segundo Stellman e Greene (2015) tal sistema representa a melhor forma de remover a desigualdade e prevenir a sobrecarga das equipas e dos seus elementos.

6. Fortalecer a equipa

“Pessoas comprometidas e que refletem sobre os assuntos providenciam uma vantagem competitiva sustentável” (Poppendieck, M. & Poppendieck, T., 2003).

Tal princípio Lean reforça a ideia de que a confiança nos membros da equipa e na sua capacidade de executar as tarefas da melhor forma incentiva-os a procurar maneira de melhorar os seus processos (Janes & Succi, 2014).

Essa prática promove o reconhecimento das equipas pelas suas conquistas e a consciência de que as pessoas são o recurso mais valioso e que não deve ser desperdiçado.

7. Otimizar o todo

Na otimização do todo reside outro ponto importante do pensamento Lean, não deixando de ser relevante a sua aplicação no contexto do desenvolvimento de *software* (Hibbs et al., 2009). A análise do todo de forma a otimizá-lo constantemente permite tomar conhecimento do que é que a equipa está a fazer de forma eficiente e eficazmente, sendo, para tanto, importante dar um passo atrás e ver o panorama total (Janes & Succi, 2014). Além disso, importa efetuar medições, pois munidas da informação certa na altura certa, as equipas entendem melhor os seus objetivos e a visão geral do projeto, podendo os gestores identificar quais os projetos que necessitam de ajuda, detetar e resolver problemas de pontos de restrição, e, bem assim, prevenir atritos e a exaustão das equipas (Martin, 2009).

Uma das medições muito comuns é fornecida pelo *Lead Time*, o qual mede o tempo que passa entre o início e o fim de uma tarefa assim que o pedido é efetuado, o que acaba por traduzir o tempo que o cliente esteve à espera de determinada funcionalidade (Stellman & Greene, 2015). Outra medida importante e utilizada para avaliar a produtividade de uma equipa consiste na velocidade de desenvolvimento, que revela o índice de entrega de valor pela equipa. A entrega de valor corresponde ao número de funcionalidades completas observadas num dado período de tempo, as quais estão prontas para produção ou testes (Martin, 2009). Assim, quando se alude à velocidade de uma equipa, normalmente refere-se à velocidade média, isto é, ao cálculo efetuado com utilizando vários intervalos de tempo para estabelecer uma tendência.

Para além das métricas, há outro modo de inventariar eventuais problemas e solucionar-los por forma a atingir a melhoria contínua, como, por exemplo, a prática dos Cinco «Whys» que ajuda a encontrar a causa de alguns problemas encobertos. Tal processo consiste em perguntar aos membros da equipa o porquê de ter acontecido um determinado problema, e quando estes respondem volta-se a colocar a mesma questão, usualmente cinco vezes até se descobrir a origem do problema (Stellman & Greene, 2015). Quando se detetam ineficiências e se opta por incidir os esforços para otimizar apenas uma parte do processo, como já foi dito anteriormente, está-se a sub-otimizar os

sistemas. O ideal é que as otimizações devam sempre conseguir o máximo de valor possível, de preferência o sistema completo para maiores benefícios. (Hibbs *et al.*, 2009).

1.5.4 Casos de Sucesso

Ericsson

De seguida apresenta-se uma visão sumária do caso de estudo desenvolvido por Mary e Tom Poppendieck em 2013, acerca da introdução do Lean IT na Ericsson.

Fundada em 1876 na Suécia, a Ericsson ganhou o estatuto de empresa fornecedora líder de infraestrutura de telecomunicações, sendo que, na última década, graças ao incremento do ritmo da mudança, o foco da sua atividade passou do *hardware* para o software. Uma tal mudança impôs-lhe como necessidade a colocação no mercado de novos produtos, de forma mais acelerada do que o que estava habituada, e com necessidade de ter previsões de entrega confiáveis.

Para diminuir o «time-to-market» os responsáveis da Ericsson propuseram a troca dos modelos tracionais, cujos projetos eram orientados à «release», para o modelo Lean e Agile centrado na velocidade, focado no desenvolvimento de funcionalidades e fortalecimento das equipas. Preconizaram, além disso, o desenvolvimento de uma transição gradual, atribuindo às equipas a responsabilidade de descobrir os detalhes do novo processo a seguir. A primeira alteração introduzida no quadro de um tal processo consistiu, por isso, na organização das equipas, que passaram a ser multidisciplinares e organizadas em torno de funções com cerca de sete elementos. Estabeleceu-se também uma lista das funcionalidades mais importantes desejadas para cada produto, cuja manutenção era obrigatória. Adicionalmente, era dada a responsabilidade completa à equipa de descobrir como deveria entregar determinada funcionalidade, sabendo, à partida, que cada funcionalidade tinha restrições de tempo, devido ao valor a ela associado e à sua necessidade no mercado.

Todo o código desenvolvido deveria ser integrado em uma base regular de modo a encontrar defeitos imediatamente, mantendo-se assim limpo sem dependências escondidas. A equipa deveria trabalhar uma funcionalidade de cada vez, operando todos os membros da equipa em conjunto, cabendo a esta a responsabilidade de encontrar um especialista adicional caso se revelasse necessário. As «*releases*» tinham de ser desacopladas das funcionalidades, deveriam ser frequentes e só as funcionalidades completas - testadas, integradas, é que seriam incluídas na «*release*».

Resultados na Ericsson

Em consequência das novas práticas verificou-se que os pedidos de alterações ou «*change requests*», que anteriormente absorviam muito tempo e energia das equipas, desapareceram. Como se seguia a máxima de ‘decidir o mais tarde possível’ passou a não haver necessidade de perder tempo a refazer os planos e os problemas associados com os relatórios internos diminuírem significativamente, porque a maioria dos problemas eram resolvidos na ‘fonte’ devido à característica de multidisciplinaridade da equipa. Além disso, o tempo de construção de uma «*build*» diminuiu drasticamente por necessidade, pois, ao contrário do que acontecia antes, quando eram feitos decorridos meses e não tinham se ser especialmente rápidas, passaram a ter de ser realizadas diariamente, o que implicou o investimento de algum tempo numa automatização de processos que permitiu fazer em minutos o que era feito em dezenas de horas.

Apesar do objetivo primordial da Ericsson assentar na redução do seu «*time-to-market*», que, com esta transformação caiu para metade, verificou-se, complementarmente, a um aumento de qualidade, detetando-se menos defeitos inclusive os reportados pelos clientes. Constatou-se, assim, ser possível executar mais trabalho, que o número de horas por funcionalidade diminuiu 50% e que as funcionalidades entregues eram as corretas e espetáveis.

O feedback dos colaboradores da Ericsson evidenciou-se também positivo em relação a esta iniciativa, declarando os ativos sentir que as suas capacidades eram melhor aproveitadas, porque o trabalho em conjunto para resolver problemas lhes permitia usar as suas capacidades totais em vez de se sentirem desperdiçados devido a uma alocação em múltiplos projetos que provocava trocas de tarefas e o reinício do processo todo de novo, vezes sem conta.

A par disso, valorizava-se mais os elementos das equipas, pois deixou se ter de se aguardar no passado pelo parecer de um especialista externo sempre que se deparava, um problema no desenvolvimento do projeto, passando, na nova abordagem, as equipas a resolver os problemas com os membros que tinham e mediante a interajuda, gerando por um sistema de reciprocidade informal.

Em resumo, entre vários princípios e práticas Lean adotados pela iniciativa da Ericsson destaca-se a aplicação da ‘eliminação dos desperdício’ com abordagem das funcionalidades ‘corretas’ na entrega do valor ao cliente, o fortalecimento da equipa, através da atribuição de poder de decisão e responsabilidades extra e o «*just-in-time*»,

sendo que os produtos passaram a chegar ao mercado a tempo da Ericsson apresentar uma boa competitividade.

Wipro technologies

O segundo caso de sucesso apresentado surge relatado numa investigação executada por Staats *et al.* (2011) e tem por referência a iniciativa Lean operações internas aplicadas pela consultora indiana Wipro Technologies e que tiveram impacto na performance empírica dos seus respetivos projetos.

A Wipro é uma empresa de tecnologia da informação, consultoria e *outsourcing* indiana de implantação internacional que agrega cerca de 170.000 colaboradores e que dispõe de clientes distribuídos por 175 cidades de seis continentes. No início de 2004 a Wipro confrontava-se com uma série de questões de ineficiência de operações em alguns dos seus projetos: muitos colaboradores assistiam a demasiadas reuniões restando-lhes pouco tempo para executar as tarefas de que eram responsáveis; o tempo à espera de resposta de terceiros para tomar decisões era demasiado longo; eram numerosos os pontos de restrição, registando-se trocas excessivas de tarefas e muitas situações de impossibilidade de entregar projetos nas datas previstas, o que provocava «burnout» desnecessário dos trabalhadores. Para resolver as limitações e ganhar vantagem competitiva a Wipro introduziu alterações ao sistema de funcionamento direcionadas ao equilíbrio de resultados e a prevenir eventuais réplicas da concorrência.

As medidas adotadas pela Wipro, baseadas num sistema Lean IT específico, viriam a contribuir, de forma direta ou indireta, para atingir um assinalável aumento de eficiência. A opção por práticas e ferramentas Lean específicas teve que ver com a necessidade de adaptação de regras para melhor servir os objetivos de cada uma das equipas, estratégia que viria a ter repercussões em várias vertentes dos projetos como se poderá perceber pelo que abaixo se refere.

Do tradicional para o Agile

No quadro da mudança protagonizada, que acabou por se traduzir em melhorias num projeto «time and materials» que Wipro tinha há já alguns anos, o respetivo gestor decidiu propôr ao cliente a abordagem Lean ensaiada na empresa, explicando-lhe as vantagens que isso poderia trazer à sua organização. O cliente aceitou

a proposta, confirmando-se, ao fim de alguns ciclos iterativos de desenvolvimento, as vantagens que a nova abordagem trazia. Constatou-se também que as pequenas demonstrações do produto no fim das iterações e sem este estar totalmente completo, provocam a abertura do pensamento e expansão das ideias, o que representa mais valor para o negócio. Verifica-se muitas vezes que os SI têm o papel de melhorar a produtividade do negócio dos seus clientes (internos ou externos) e até de contribuir para uma melhor clarificação da estratégia empresarial. Nesta caso particular verificou-se que a abordagem Lean IT não foi apenas eficiente para o IT, sendo-o também para o sector de negócio.

Especificação de Tarefas

De todos os princípios Lean aplicados nos projetos da Wipro, a especificação de tarefas, no âmbito na necessidade de expansão do conhecimento, constitui o que trouxe menos melhorias em termos globais. No entanto, alguns dos projetos onde se aplicou este princípio revelaram-se bem-sucedidos devido à padronização de práticas e tarefas. São disso exemplo os projetos que tiveram de ser replicados mais do que uma vez, em clientes e/ou países distintos, com apenas alguns ajustes e configurações de acordo com o contexto. A standardização e especificação de tarefas mostrou que o esforço despendido na primeira implementação não tinham de ser replicado à medida que novas implementações do mesmo projeto iam sendo necessárias. O esforço aplicado viria a ser compensado com a produção de um número menor de defeitos, uma menor quantidade de trabalho a refeito, ou seja no global, registou-se aumento de produtividade.

Matriz Estrutura do Projeto

A introdução da matriz estrutura do projeto (*DSM – Design Structure Matrix*), uma ferramenta Lean muito utilizada, ajudou a tornar a comunicação mais direta entre as partes envolvidas e a simplificar a arquitetura.

As entradas da DSM constituem atividades e/ou tarefas do projeto e os *outputs* correspondem às suas dependências, as quais permitem a dedução de uma forma de ordenar cronologicamente as tarefas listadas. Com a DSM quem planeia pode verificar conflitos futuros, através de dependências, e pode potencialmente resolver problemas

arquiteturais pois a DSM, segundo os autores do estudo, reduz a ambiguidade e aumenta a visibilidade do processo como um todo.

O aspeto positivo desta ferramenta em relação à gestão tradicional está na qualidade de assegurar ao gestor do projeto a possibilidade de extrair o verdadeiro conhecimento dos ativos e de colocá-lo em papel para poder ser disponibilizado, em vez de o forçar a planear mais e a observar melhor.

Quadro de Controlo de Produção

A introdução dos Quadros de Controlo de Produção, «*Visual Control Boards*» VCB, viriam, por outro lado, a contribuir para identificar processos e problemas anteriormente com pouca ou nenhuma visibilidade, sendo este um aspeto difícil da aplicação *lean* ao conhecimento. O VCB facilita a comunicação direta, pois torna evidente em termos visuais, as dependências entre os vários membros da equipa e suas tarefas. Em caso de qualquer membro da equipa ter alguma questão sobre o *output* da sua tarefa ou estiver à espera de um *input* para poder prosseguir o trabalho, poderá dirigir-se o quanto antes à pessoa devida, pois tem visibilidade de quem se trata. O VCB representa também o único ponto de situação do projeto, centralizando os problemas, e de certa forma, fazendo um autodiagnóstico dos processos visto que utiliza os *inputs* dos membros da equipa e acompanha a respetiva evolução, indo ao encontro dos objetivos de cada membro. A filosofia dos colaboradores da Wipro jamais foi de pedir ajuda, optando, normalmente, os membros das equipas por esperar pelos pontos de situação de maior dimensão para identificar possíveis problemas. Porém, a partir do momento que começaram a aderir ao VCB os próprios membros da equipa, tal como o gestor do projeto, passaram a poder identificar de forma sistemática as questões e a endereça-las através da procura de auxílio ou com recurso a formações cada vez mais precoces.

Enquanto DSM divide o projeto em módulos e ajuda a criar uma ordem inicial para as tarefas, o VCB permite rastrear dinamicamente os processos, devendo ser usado durante a execução de um projeto. O gestor de projeto fica responsável por preencher o VCB para uma ou duas semanas de cada vez. Assim, se as circunstâncias do projeto se alterarem devido à incerteza das tarefas, o VCB deve ser alterado do modo a estar

sempre atualizado e a refletir a realidade. Mais uma vez, se o VCB estiver atualizado, o gestor de projeto identifica os problemas e soluciona-los atempadamente, detetando, por exemplo, alguma sobrecarga na equipa ou, pelo contrário, apercebendo-se se os membros da equipa terminaram o trabalho mais cedo para poder realocar o trabalho de forma mais eficiente.

Mapa de Fluxo de Valor

Numa das equipas envolvidas neste caso de estudo - que estava responsável por resolver defeitos identificados obedecendo a um determinado SLA - foi aplicado o VSM aos respetivos processos. Na fase inicial procedeu-se a um levantamento dos processos, identificando-se os que traziam valor e os outros que, pelo contrário, se consideravam desperdício. O processo rastreado iniciava-se no momento em que um defeito era identificado e terminava quando se confirmava a sua resolução. Constatou com tal exercício que muitas das vezes os programadores tinham de esperar pelos «testers» para avançar com o seu trabalho porque se gerava um ponto de restrição no lado da verificação das correções. Depois de se examinar a VSM os responsáveis da Wipro decidiram atribuir a tarefa de testes também aos programadores e tornar mais direta a comunicação entre estes e os «testers». Além disso foi identificada igualmente a necessidade de se eliminar todo o código redundante para diminuir o número de defeitos. Por essa via, a equipa passou de 1.3 defeitos para dois (por programador) corrigidos em cada semana, o que resultou numa diminuição do SLA e da qualidade do sistema.

Fluxo contínuo

O Lean defende a criação de um fluxo contínuo de forma a eliminar a pressa excessiva na execução das tarefas e também o seu oposto, a execução com folga em demasia, tendo uma das equipas que aplicou diretamente este princípio verificando-se uma mudança significativa na qualidade do *software*. Antes das alterações introduzidas, os colaboradores eram forçados a terminar uma tarefa o mais rapidamente possível, constatando-se que ao terminar a tarefa precocemente se tornava necessário esperar para poder entregar o *software* ao cliente, perdendo-se os ganhos de se ter executado a tarefa com elevada velocidade. Ao aplicar o princípio do fluxo contínuo, a equipa formulou

uma nova arquitetura para o processo fazendo com que se distribuísse o esforço dos testes ao longo de todo o desenvolvimento e, com isso, promovendo a continuidade do fluxo, em vez de os concentrar todos no fim. Tal estratégia traduziu-se em poupança de recursos humanos e ganhos de tempo suficientes para acomodar novos pedidos de alteração feitos pelo cliente.

Modelo Iterativo

O desafio real da aplicação do *lean* no conhecimento (em contraste com a produção) reside na circunstância da maior parte do trabalho efetuado não ser facilmente e, não sendo possível estar continuamente a avistar os problemas, revela-se difícil endereçá-los atempadamente e conseqüentemente haver uma adaptação. Além do mais, as hipóteses formuladas erradamente são muito mais difíceis de identificar, o que torna fundamental a aplicação de métodos iterativos. A tarefa de identificar problemas atempadamente num *software* não é simples, sendo que no fim do desenvolvimento a dificuldade decai, mas nessa fase o processo de correção revela-se muito dispendioso.

Antes da adoção das novas práticas, a Wipro poderia levar semanas ou meses a identificar um defeito, agora tudo se faz rapidamente diminuindo-se o esforço da execução de tarefas por mais do que uma vez, precavendo-se os membros das equipas contra a repetição do mesmo erro noutras situações.

De forma conclusiva poderá afirmar-se que, para além dos evidentes aumentos de produtividade e melhoria na qualidade de *software* produzido o que constituiu um importante passo para o sucesso da Wipro, a adoção do Lean IT pela empresa permitiu uma grande redução de incerteza em relação aos seus maiores e complexos projetos e à conquista de uma maior confiança.

2 Abordagem de Investigação

Uma investigação do tipo daquela que aqui se propõe envolve a utilização de dados qualitativos, como entrevistas, questionários, documentos e textos, além de informação resultante da observação direta de participantes em projetos e da interpretação do próprio investigador, cuja recolha e seleção se tornam necessários à explicação e o entendimento do fenómeno em estudo.

Como método de pesquisa adotou-se, por conseguinte, o método de caso de estudo, entendido como adequado à investigação na área dos sistemas de informação por que avaliar os sistemas de informação das organizações assim como os problemas organizacionais a eles inerentes. Yin (2002) define o caso de estudo como: *“Investigação de um fenómeno contemporâneo dentro do contexto da vida real, especialmente adequado quando as fronteiras entre o fenómeno e o contexto não são claramente evidentes”*.

Para atingir os objetivos e responder às questões da investigação aqui desenvolvida impôs-se como primeiro passo a realização de uma revisão da vasta literatura, a qual incluiu a alusão a alguns casos reais publicados sobre gestão tradicional de projetos e gestão Lean IT em livros e artigos de revistas da especialidade. Por essa via, procurou-se construir uma base teórica de conhecimento que permitisse enquadrar e analisar de forma neutra os casos de estudo sobre os quais se debruça este trabalho.

Concluída a elaboração do estado da arte, revelou-se proveitoso organizar todo o material disponibilizado pelas empresas de forma a fazer uma pré-seleção de documentos que proporcionassem um entendimento conducente à formulação de respostas às questões em investigação.

No caso A, assumiu-se como essencial a análise dos cronogramas do projeto, tanto o estimado inicialmente como as suas sucessivas adaptações. Para tanto, consultaram-se atas de reuniões entre cliente e prestador de serviços IT bem como relatos sobre reuniões internas da empresa. Revelou-se também necessário analisar o conteúdo de documentação de análise dos requisitos, desenho técnico e documento de *kick-off* do projeto de desenvolvimento, além dos relatórios de execução de testes de

integração e de aceitação com o cliente. De referir, por exemplo, que da apreciação dos relatórios de progresso do projeto examinados dependia a apreciação do detalhe sobre os riscos respetivos. O progresso das tarefas da equipa foi documentado no ficheiro com o EDT, o qual, ao não ser atualizado com a frequência expetável, tornou impossível a concretização do propósito de obtenção de indicadores de desempenho suficientemente completos para documentar o caso. Os dados conseguidos limitam-se, por isso, aos que serão apresentados na descrição do caso de estudo, cuja limitação para efeitos de uma ilustração completa, se reconhece.

No caso B foram apreciaram-se documentos empresariais de que constam exemplos do planeamento semanal, exemplos de preenchimento do *Whiteboard* utilizado pelas equipas, planos mensais e trimestrais bem como documentos de suporte a reuniões de feedback e documentos de suporte a ações de melhoria. Adicionalmente consultaram-se os dados referentes aos KPIs de desempenho das equipas ao longo dos anos de aplicação Lean IT bem como os do ano prévio à sua introdução (2013, 2015 e 2010 respetivamente).

Este trabalho constitui uma investigação qualitativa, resultando na execução de uma narrativa baseada em observações interpretadas e sumarizadas tendo em conta a escala e dimensão do objeto da pesquisa.

Objetivos de investigação

A investigação, como referido anteriormente, teve por propósito compreender se a produtividade das equipas que executam projetos em IT acusa influência direta ou indireta em resultado de introdução da abordagem ou *pensamento* Lean quando aplicado às tecnologias de informação. Complementarmente, pretendeu-se entender se a mudança para a nova forma de organizar e realizar tarefas se refletiu na qualidade do *software* produzido. De modo a clarificar estas questões irá recorrer-se ao caso de estudo B.

Enquanto a narrativa desenvolvida em torno do caso A visou tão só ilustrar o modo de funcionamento de um projeto tradicional, tipicamente situado no extremo oposto ao dos projetos denominados Agile ou Lean, o relato produzido em referência ao caso B teve o propósito de responder às questões colocadas pela introdução dos princípios e das ferramentas Lean na produção de software.

3 Caso de estudo

No capítulo anterior procedeu-se a uma revisão de literatura sobre as metodologias tradicionais de gestão de projeto a par das consideradas como mais ágeis. Neste capítulo pretende-se pô-las em confronto, através do tratamento de dois casos de estudo: um projeto em que são adotadas técnicas e as práticas da metodologia convencional e outro em que são seguidas orientações que configuram a aplicação do Lean IT, considerada uma vertente da metodologia “leve”.

3.1 Análise do caso A – Gestão tradicional de projetos

No caso A, está em causa a apreciação de um projeto de desenvolvimento feito à medida por requisição de uma entidade de regulação bancária internacional (o BNI) a um fornecedor de *software* que atua no mercado português. Para atingir com sucesso os objetivos que se propunha a organização (fornecedor) adotou a abordagem de gestão linear, ou, como é comumente designada, gestão tradicional de projetos, considerada das mais simples e intuitivas (Wysocki, 2006) e adotada quando o objetivo é perfeitamente conhecido e se espera que a solução encontrada conduza ao efeito esperado. A estratégia adotada baseia-se, portanto, no pressuposto de não acomodar facilmente desvios. À parte disto, não pretende o projeto em estudo ser uma referência ao que deve ser o modelo tradicional, tanto mais que, muitos dos seus aspetos foram simplificados para destacar suas componentes mais relevantes e não alongar demasiado este trabalho de investigação.

3.1.1 Enquadramento

O BNI, Banco Nacional Internacional⁶ (entidade a quem se destina o projeto em análise) desempenha funções de regulação, competindo-lhe a missão de garantia da estabilidade do sistema financeiro, supervisionando, para tanto, as instituições

⁶ Nome fictício para proteção da confidencialidade do cliente deste caso de estudo.

financeiras e não financeiras de acordo com o disposto na lei do país onde atua. Surge assim incumbido de tarefas de determinação e fiscalização do cumprimento de todas as relações prudenciais que as Instituições Financeiras (IFs) devem observar para garantir a respetiva solvabilidade e liquidez. Além disso, segundo a legislação aplicável, enquanto supervisor, o BNI é também responsável por executar inspeções regulares *on-site* às IFs, no contexto das quais deve emitir recomendações e acompanhar a implementação respetiva.

Para assegurar o cumprimento das exigências impostas por lei, o BNI desenvolveu uma metodologia de supervisão baseada no risco e adaptada ao seu sistema financeiro do país, tendo como referência o que já foi implementado no plano internacional, em obediência às orientações do Comité Basileia, designadamente os *Core Principles*. Visando assegurar o alinhamento com esses objetivos, o Departamento de Supervisão das Instituições Financeiras, de agora em diante referido por DSI, definiu, conjuntamente com uma consultora estratégica, o detalhe e os moldes da metodologia, estabelecendo, de igual modo, os procedimentos, as ferramentas a utilizar e o trabalho a desenvolver pelas equipas de supervisão no dia-a-dia. O modelo de supervisão teórico desenhado inclui a listagem de normas e processos, culminando com o desenvolvimento de um protótipo em VBA destinado a dar suporte informático e a centralizar todas as atividades de um supervisor. A referida ferramenta, denominada AIF Tático (Aplicativo das Instituições Financeiras – Tático) permite detalhar os procedimentos e recomendações dos supervisores efetuados *on-site* aquando a sua deslocação a uma IF para inspeção e a análise qualitativa e quantitativa dos vários indicadores de risco das IFs, calculados após a submissão mensal e obrigatória por regulamentação do balancete das IFs ao BNI. A AIF Tático releva-se, porém, ineficaz, pois em poucos meses de utilização a ferramenta tornou-se um sistema de baixa performance, devido às limitações tecnológicas associadas às tecnologias sobre que fora desenvolvido. Os supervisores queixavam-se, por exemplo, da perda de muito tempo quando tentavam aceder à informação que tinham introduzido em cada instituição. Mas essa não se revelava a maior falha identificada: a principal desvantagem do AIF Tático consistia no facto de, por não estar integrado com os sistemas de informação do BNI, obrigar os supervisores a ter de carregar, mensalmente, a informação do balancete atualizado por cada uma das instituições que supervisionavam de forma a avaliar os indicadores de risco. Tinham até, em algumas

situações, de o fazer mais do que uma vez, pois as IFs, esporadicamente remetiam segundas vias corretivas do mesmo balancete no decorrer do mês. Para desespero dos supervisores e dos seus responsáveis a tarefa de extrair os relatórios mensais e trimestral de risco era morosa, sendo a responsabilidade dos atrasos a endereçada em exclusivo a uma ferramenta de baixa performance e má usabilidade.

Para resolver os problemas listados, o DSI relatou a sua situação ao Departamento de Tecnologias de Informação do BNI, que, em face do impacto gravoso da situação na garantia do cumprimento das responsabilidades associadas ao exercício, deu nota à Direção, constatando-se a instituição não dispor de recursos humanos qualificados com vista ao desenvolvimento do AIF integrado. Consequentemente, o BNI decidiu recorrer a uma consultora de tecnologias de informação, a ConsultIT⁷, para desenvolver a nova ferramenta - o AIF Evolutivo - em colaboração com a área de negócio

3.1.2 Descrição do âmbito inicial do projeto

O objetivo do projeto aprovado consistia na criação de uma nova aplicação que deveria estar incluída no Portal das Instituições Financeiras do BNI em que já existiam outras aplicações de suporte à supervisão, nomeadamente a de exploração de dados com BI (Oracle). O AIF Evolutivo deveria dispor da base de dados comum à plataforma já existente (após o processo de ETL) onde eram armazenados todos os dados reportados pelas IFs, fossem eles mensais, quinzenais e diários, todos eles obrigatórios por regulamentação. A atualização frequente e automática representava uma exigência fundamental para o AIF Evolutivo visto que, sem isso, o acompanhamento contínuo das instituições através inspeções *off-site* ficaria comprometido. Assim, o AIF Evolutivo seria concebido para integrar componentes a alto nível (especificadas na tabela), baseadas nas já constantes do em utilização:

⁷ ConsultIT consultora internacional com sede em Portugal atua na área de tecnologias de informação, nome fictício de forma a proteger a confidencialidade da prestadora de serviços.

Bloco Funcional	Nº de Requisitos	Descrição
Acessos e autenticação	10	Reaproveitamento das funcionalidades de autenticação já existentes do portal base e criação de novos perfis de utilização do AIF.
Ecrã de Transição	5	Permite a escolha da instituição financeira a consultar/editar;
Menu de Navegação	13	Menus de topo de laterais que permitem navegar em toda a aplicação;
<i>Dashboard</i> Inicial	28	Apresentação e caracterização da Instituição financeira bem como avaliação atribuída em relação ao risco;
Alertas	157	Componente alarmística responsável por chamar a atenção aos supervisores quais os indicadores de risco assim que ultrapassarem os limites regulamentares impostos;
Ficha institucional	37	Apresentação dos dados que caracterizam a instituição e o seu exercício, bem como as recomendações prudenciais a estas prestadas;
Planeamento	40	Possibilidade de criar um plano detalhado de atividades de inspeção <i>off-site</i> de acordo com a metodologia de supervisão implementada;
Inspeções	37	Possibilidade de criar um plano detalhe de atividades de inspeção <i>on-site</i> de acordo com a metodologia definida; (dentro do âmbito inicial mas não implementado. Para mais detalhe ver a secção Análise e Desenho Funcional.)
Documentos relevantes	50	Área associada à documentação de entrega obrigatória ou adicional que deve ser entregue pelas IFs ao BNI e que tem alarmística associada às datas limites impostas.
Dashboard de Notas	100	Consulta das notas previamente atribuídas e aprovadas para determinada instituição financeira;
Avaliação de Risco	731	Possibilidade de avaliar vários aspetos dentro das categorias de forma a determinar o grau de risco global da IF. Permite a consulta de aproximadamente 45 indicadores de risco dentro das categorias de avaliação: Risco de Crédito, Risco Mercado, Risco de Liquidez, Risco Operacional, Organização e Gestão, Solvabilidade e Liquidez.
Comentários	248	Resumo dos comentários executados pelo supervisor

gerais		aquando da sua revisão detalhada dos aspetos e categorias sobre avaliação. Recomendações às instituições financeiras.
Documentos de Supervisão	413	<p>Extração de relatórios dinâmicos e documentos estáticos no formato .docx:</p> <p>Relatórios dinâmicos:</p> <ul style="list-style-type: none"> • Relatório Sumário • Relatório Detalhado • Relatório de Notas • Relatório de Inspeção <p>Documentos estáticos:</p> <ul style="list-style-type: none"> • Carta de Notificação • Carta de Recomendações • Carta de Informação Interna de Infração
Backoffice	215	<p>Configuração e parametrização de vários aspetos do AIF Evolutivo, como a gestão de IFs, de divisões e setores. Parametrização de ponderadores, configuração do mapa de atividade, gestão de documentos obrigatórios e ad-hoc, configuração de atividades de supervisão e passagens automáticas de ciclos de supervisão.</p>
Nº total de requisitos planeados		2084
Nº total de requisitos implementados		2047

A *baseline* para este projeto considera dentro do seu âmbito todos os requisitos anteriormente incluídos na ferramenta prototipada, o AIF Tático. Além disso, adiciona-se, como é natural, alguns requisitos mínimos extras para suporte de um sistema de informação integrado, com boa performance como o que o AIF pretendia ser.

3.1.3 Avaliação técnica do projeto

Revela-se de extrema importância proceder a uma primeira avaliação dos riscos do projeto, mesmo antes do seu arranque. O objetivo desta avaliação inicial reside

na conveniência de perceber a capacidade da organização no momento para dar resposta à construção do sistema proposto. Se todos os projetos têm riscos, ainda que numa fase muito inicial muitos deles assentam na avaliação das opiniões do grupo que a está a efetuar (Miguel, 2015), há necessidade de tomar consciência das fontes dos riscos para atingir em plenitude os benefícios esperados.

A dimensão e complexidade do sistema

Na perspetiva de Rakos (1990) um projeto passa a ser considerado de grande dimensão quando abrange 100 person-years⁸, sendo que o ideal é ter-se cinco a sete pessoas trabalhar até se solucionar o problema. Ora, no presente caso de estudo o número de elementos da equipa chegou a ser aproximadamente 14 elementos, embora quatro tivessem intervenções muito reduzidas e pontuais em fases de *congestionamento*. Pode-se então considerar-se que integraram em permanência a equipa dez elementos dedicados, o que se traduza numa relação de 10 person-years (tempo real) e 12,7 person-years (tempo estimado), o que, segundo Rakos (1990), remete o projeto claramente à escala de pequena-média dimensão. (Tempo de duração do projeto - 9,5 meses, tempo estimado - , 12 meses tempo real.)

O número de departamentos organizacionais envolvidos no projeto - dado que funciona como indicador do grau de complexidade das relações que devem ser mantidas: três - Departamento de Supervisão das Instituições, Departamento de Regulação e Organização do Sistema Financeiro e Departamento de Tecnologias de Informação. Trata-se, assim, de um projeto de desenvolvimento paralelo, comportando um fator externo não controlável pela gestão, o qual estava a adaptar as normas internacionais de relato financeiro (IAS/IFRS) com consequências na adaptação do sistema de informação, fundamental para o aplicativo do âmbito a implementar. Outro fator externo que importava considerar tem a ver com a dependência da informação e expertise de consultores exteriores à organização.

As características inerentes ao projeto ajudam a prever de antemão parte do risco associado, sendo relevantes considerar-se se é um projeto completamente novo ou se é um projeto de adaptação de um sistema já existente, associado a diferentes níveis de complexidade. No corrente caso em estudo, o projeto pretende ser completamente novo,

⁸ Person-years/weeks/days ou hours é uma medida de esforço e representa o tempo total que a pessoa trabalhou no projeto.

embora tenha por referência um protótipo pré-existente, desenvolvido essencialmente em EXCEL (com VBA) e sem integração com a fonte de dados.

Um dos fatores passíveis de influenciar o sucesso do projeto reside na perspetiva de alteração de hábitos e procedimentos do quotidiano imposto aos utilizadores. Previu-se, assim, a introdução de mudanças nos procedimentos do dia-a-dia do trabalho dos utilizadores finais, que passariam a deixar de extrair mensalmente os dados relativos à informação dos balancetes mensais das instituições financeiras, atividade a que estavam habituados quando utilizavam o sistema antigo. Ao mesmo tempo que se trabalhava no desenvolvimento do novo sistema de informação, outra consultora estratégica ocorria, aludida anteriormente: a implementação de um novo modelo – o Modelo de Avaliação das Instituições Financeiras - o qual implicou a alteração de muitos procedimentos e organização do departamento do DSI.

Quanto à perceção e vontade dos utilizadores em participar verificou-se que se mostravam-se disponíveis para responder a questões quando lhes eram direcionadas presencialmente, mas sempre que contactados por *email* ou à distancia as respostas demoravam, não sendo detalhadas quanto necessário.

Equipa e Responsabilidades

Designação, competências e atribuições dos membros da equipa do projeto são apresentados seguidamente com uma breve descrição das suas funções e responsabilidades.

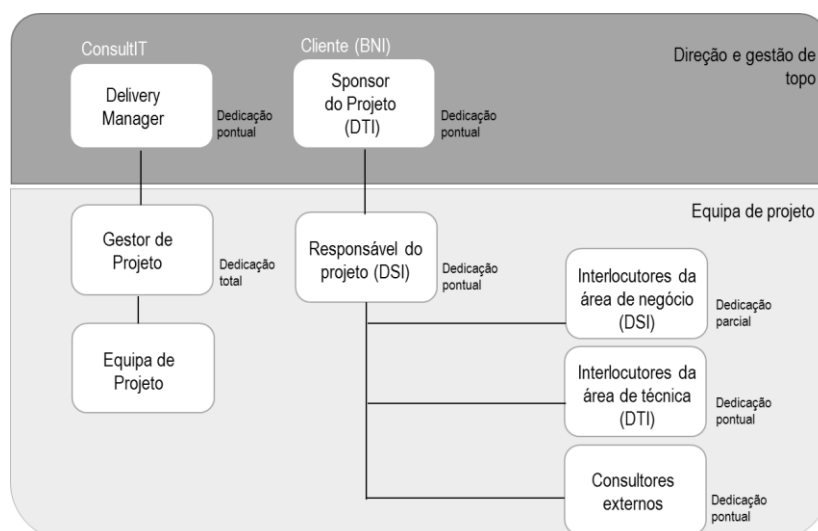


Figura 5: Estrutura organizacional dos envolvidos no projeto AIF

Ao IT Delivery Manager incumbe, como principal tarefa, a submissão faseada de entregáveis ao comité de *steering*, de forma a obter a aceitação, aprovação e financiamento do negócio ou do IT (Bailey, 2004).

O Sponsor do projeto ou patrocinador, tem a responsabilidade de apoiar o projeto dentro da organização, podendo ser o diretor, que tem o poder de efetuar os pagamentos, ou um gestor que reporta à direção. O importante é que assuma o projeto em todas as suas vertentes e que possa ter meios para garantir os recursos quando necessário. Também participa no comité de *steering*.

Ao gestor do projeto compete a direção e coordenação da equipa de projeto, assumindo a responsabilidade pelo desenvolvimento do plano de projeto e a garantia da sua execução. Incube-lhe, ainda, a defesa do contrato e das cláusulas que definem o que deve ser ou não implementado.

O responsável do projeto assegura a direção operacional do projeto, devendo facilitar a articulação entre todas as entidades intervenientes e garantir a agilização dos contactos necessários com os elementos chave da organização.

Aos interlocutores da área de negócio cabe geralmente responsabilidade de fornecer, com recurso por exemplo a entrevistas, informações sobre o negócio confiáveis e em tempo útil. No caso de estudo, constatou-se que apesar das equipas da ConsultIT se deslocarem à sede do cliente, nem sempre era possível a obtenção de

esclarecimentos, visto que os supervisores se encontravam, por vezes, em formação ou a desempenhar tarefas urgentes e não planeadas para a direção.

Os interlocutores da área técnica têm por missão disponibilizar informações sobre os sistemas de informação da organização acerca das práticas e *standards* comuns aos quais o novo sistema deverá ter de obedecer e assegurar a validação técnica e conformidade da solução aquando a entrega final.




Compete aos consultores externos prestar auxílio aos interlocutores da área de negócio a fim de poderem contribuir de forma mais otimizada na construção da nova aplicação. Dado que os utilizadores de negócio não tinham qualquer experiência em participar e providenciar *inputs* para um projeto de desenvolvimento de software, os consultores externos foram uma mais-valia neste processo.

A equipa do projeto está, por seu lado, incumbida da análise e sistematização de toda a informação necessária recolhida em *workshops* e documentos, a qual se revela essencial para atingir de forma satisfatória os compromissos. Deve também conceber funcionalmente a solução, desenvolver o respetivo desenho técnico e reunir documentação associada. Compete-lhe ainda programar as várias componentes da solução de *software* e testá-las, além de desenvolver atividades e testes de integração e participar nos testes de aceitação com o cliente. Tem ainda a seu cargo as tarefas de apoio à entrada em produção do novo sistema e reporte do estado e as dificuldades que se deparam ao longo do processo ao gestor de projeto.

A experiência da equipa de projeto e o à vontade com que lida com as tecnologias utilizadas no desenvolvimento podem influenciar o seu desempenho ao longo do projeto. Equipas cujos indivíduos são seniores em maior número mais rapidamente conseguem assumir responsabilidades, sendo um à vontade suficiente para tomar decisões. Em contrapartida, equipas formadas por indivíduos menos experientes, evidencia a constante necessidade de recuso a orientadores, precisando de sentir a presença de uma estrutura sólida com os passos a tomar bem definidos para sentirem um maior à vontade e confiança.

Baixa		Média		Alta	
Função	Experiência	Função	Experiência	Função	Experiência
Designer UI/UX	8 anos de experiência em tecnologias similares	Programador backend/analista	2 anos de experiência em tecnologias similares	Gestor de Projeto	8 anos de experiência na função na área de negócio
Analista funcional/tester	3 anos de experiência em funções similares	Programador backend	7 anos de experiência (participação pontual)	Arquiteto de Sistemas	8 anos de experiência na função
Programador backend e SOA	2 anos de experiência em tecnologias similares			Delivery Manager	4 anos de experiência na função
Programador SOA	2 anos de experiência em tecnologias similares				
Programador frontend 1	4 anos de experiência em tecnologias similares				
Programador 2 frontend (estagiário)					
Programador 3 frontend (estagiário)					
Programador 4 frontend	4 anos de experiência (participação pontual)				
Tester (estagiário)					

Legenda:

-  Elevada experiência na área de negócio
-  Experiência média na área de negócio
-  Baixa experiência na área de negócio

A experiência do grupo de utilizadores

Os técnicos de supervisão das instituições financeiras, utilizadores finais da aplicação no caso de estudo, não possuíam experiência de participação em projetos de desenvolvimento desta natureza, embora evidenciassem familiarização com a ferramenta-protótipo que a nova pretendia substituir, conhecendo bem a componente de BI e a exploração de dados que servirá de fonte para alguns dados que a aplicação irá usar por base.

A facilidade de obtenção dos requisitos

A maior parte dos requisitos necessários à aplicação requerida revelava-se de acesso fácil e com um eficaz índice de estruturação, circunstâncias resultantes do trabalho pré-existente de suporte à ferramenta que pretendia substituir. A equipa de desenvolvimento não teve acesso à ferramenta do cliente, sendo-lhe apenas disponibilizado um conjunto de prints (ou *screenshots*) e o manual de utilizador. Por outro lado, devido à implementação do novo modelo e decorrente da utilização diária da ferramenta a substituir (também ela recente) os utilizadores sugeriam a introdução na

nova aplicação de alterações e melhorias. Tais sugestões (na prática novos requisitos), alimentadas pelos problemas detetados na aplicação em uso, trouxeram problemas adicionais, pois as melhorias apontadas surgiam com frequência, sendo assumidas individualmente pelos utilizados e algumas delas expressando opiniões opostas ao que a aplicação deveria ser.

Tecnologias utilizadas e ferramentas de suporte

As tecnologias usadas para o desenvolvimento do projeto respeitante ao caso em estudo são de utilização generalizada a nível mundial, encontrando-se bem padronizadas, como se poderá ver em detalhe na seção de desenho técnico e desenvolvimento. Apesar disso, apenas uma parte reduzida da equipa envolvida possuía o à vontade suficiente trabalhá-las em sede do projeto.

No que se refere a ferramentas de apoio a utilizar e pensadas para acelerar alguns dos processos de desenvolvimento do projeto adotou-se o Tortoise SVN para gestão controlo de versões de código, permite voltar temporariamente a uma versão anterior do código, caso se encontre um defeito numa versão que o introduziu, sendo fácil efetuar comparações entre dois ficheiros.

O SoapUI pretende simular situações de teste de carga aos serviços SOA, simulando teoricamente situações muito específicas, a maioria das quais nem se suporia que acontecessem num cenário produtivo, como por exemplo, o acesso de centenas de utilizadores ao sistema, edições e eliminações. O HP ALM Quality Center como repositório de todos os casos de testes, baterias de execuções, listagem de defeitos encontrados, revela a grande vantagem de poder medir o progresso das atividades de testes e de manter de forma centralizada todas as incidências detetadas sem perder o rasto a nenhuma inconformidade do sistema. O gestor de projeto utilizou, ainda, o Project (Microsoft), para desenhar e manter o cronograma de projeto atualizado.

Analisando os quatro fatores que influenciam o risco do projeto pode avaliar se o risco do projeto nesta caso de implementação em concreto

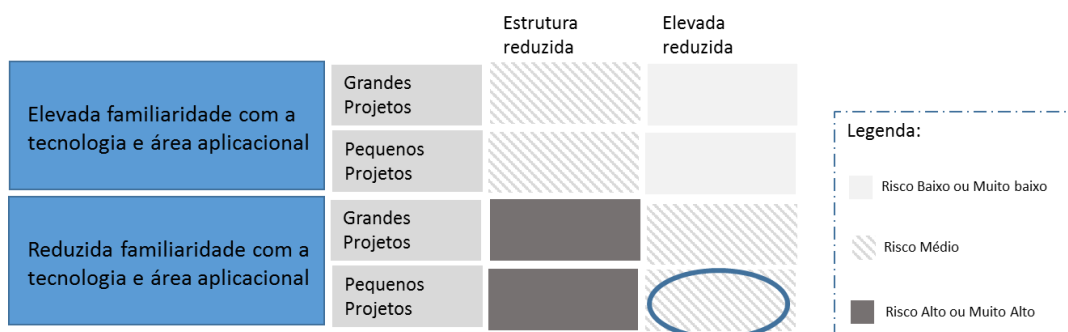


Figura 6: Matriz de Avaliação de Riscos (Adaptado de Miguel, 2015)

O projeto é por então classificado com risco médio segundo a abordagem de Miguel (2015).

3.1.4 Planeamento

O plano ou cronograma do projeto

Enquanto núcleo do plano do projeto (Stellman e Greene, 2005), o cronograma é utilizado pelo gestor de projeto para comprometer a equipa e mostrar à organização como é que o trabalho vai ser realizado. Serve também para comunicar as data finais de entrega e em alguns casos para determinar necessidades e recursos.

A forma mais comum de apresentar o cronograma do projeto recorre ao diagrama de *Gantt*, sendo cada tarefa representada por uma barra e as dependências entre tarefas por seta (como se pode ver na Figura 7). Os losangos representam milestones ou tarefas sem duração (como no exemplo, a passagem a produção executada em poucas horas).

Uma boa estimativa inicial revela-se fundamental por permitir construir toda uma base para o bom desenvolvimento do projeto. Quando confrontado com um prazo não negociável para um projeto, como é o exemplo já referido deste caso de estudo, muitos gestores de projeto trabalham de trás para a frente, partindo do prazo final de entrega para determinar o trabalho que precisa ser realizado, como foi o caso do gestor de projeto deste primeiro caso em estudo.

Seja qual for a divisão de tempo que o gestor possa fazer esses valores estão longe de resultar de qualquer conhecimento específico do trabalho exigido,

expressando, em vez disso, a necessidade de o encaixar num intervalo de tempo predeterminado.

Importa ter em consideração que o cronograma do projeto deve ser conciliado com as necessidades da organização. No presente caso, por exemplo, devido ao atraso dos desenvolvimentos e à indisponibilidade da passagem de produção, assim que possível, no fim do ano de 2016, o cliente preconizou que se deveria deixar que o sistema que introduziu as alterações ao plano contabilístico atingisse o nível máximo de estabilidade, entendendo, por isso, como aconselhável só fazer uma passagem a produção passados quatro meses após a entrada em produtivo o SI com o novo plano contabilístico IAS.

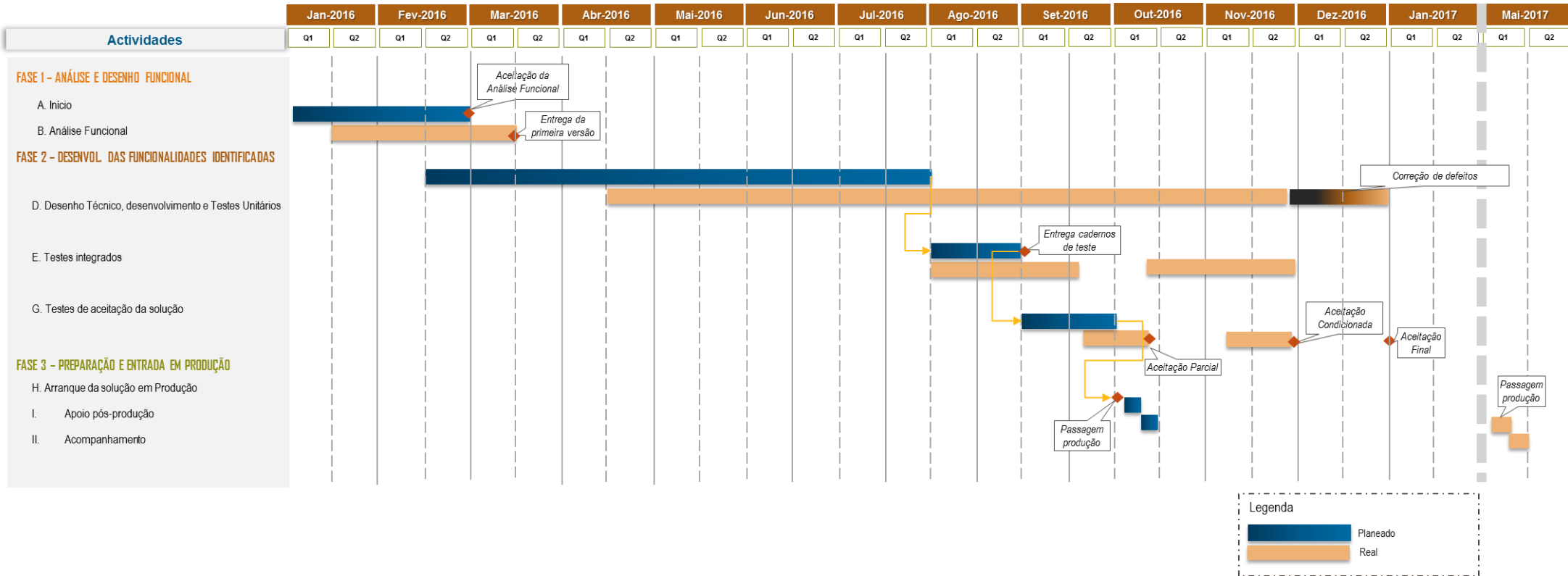


Figura 7: Cronograma do projeto AIF (estimado e real)

Relevante é, também, a definição dos entregáveis, geralmente associados a milestones do projeto, assumindo-se como fundamental o alinhamento de expectativas em relação aos entregáveis e as suas datas.

No início do projeto em estudo, foi contratualizado, após obtenção de consenso entre as partes interessadas, que os entregáveis do projeto deveriam ser os seguintes, com a respetivas datas associadas às suas milestones:

- Documento de análise funcional; 01-03-2016
- Aplicação desenvolvida para testes de aceitação; 01-09-2016
- Plano e estratégia de testes: 01-09-2016
- Caderno de testes 01-09-2016
- Manual do utilizador 30-09-2016
- Passagem a produção: 10-10-2016

Ao aceitar o plano, a gestão da parte do cliente compromete-se a aprovar a documentação associada a cada etapa para que se possa avançar para a seguinte. Nesse momento inicial ocorre, igualmente, o financiamento preliminar, sendo assumidas as decisões sobre as aprovações dos entregáveis, cedo no ciclo de vida do projeto, mesmo antes que todos os requisitos sejam conhecidos. O objetivo das estimativas preliminares reside no endereçamento dos requisitos a um nível de precisão de -25% a +75%, estimando-se, com base nisso, o orçamento do projeto.

Restrições e pressupostos do projeto

A restrição mais significativa com que se defronta o projeto em análise resulta da localização em países diferentes da equipa de desenvolvimento e o cliente/utilizadores do negócio. Várias investigações realçam que o afastamento agrava a dificuldade nas comunicações entre as equipas do cliente e a equipa de desenvolvimento Wysocki (2006). Além dessa restrição desde cedo assumida pelos responsáveis, o projeto em questão enfrentou limitações rígidas de tempo e de orçamento, contratualmente definidos *à priori*.

3.1.5 Monitorização e controlo do projeto

3.1.5.1 Estrutura de decomposição de trabalho

Antes de elaborar o cronograma do projeto, o gestor deve dispor de um EDT (Estrutura de Decomposição de Trabalho) que consiste num conjunto de estimativas de esforço de desenvolvimento para cada tarefa e de uma lista de recursos com disponibilidade para as executar.

Como a estimativa inicial do esforço do projeto foi determinada antes do seu início, por via do estabelecimento da data da respetiva conclusão, a estrutura de decomposição de trabalho desenhou-se à *posteriori*, o que não é o aconselhado como anteriormente se assinalou. Mesmo assim, o gestor de projeto trabalhou junto da equipa de forma a estimar o EDT, utilizando um dos métodos existentes baseados no consenso. Tal opção permitiu diminuir o erro da estimativa das tarefas efetuadas por alguém que vai executar o projeto ou talvez já a tenha executado em momentos anteriores. Graças à sua experiência, o gestor de projeto dispõe também do feedback necessário para se aferir da eventual necessidade de alocar recursos adicionais para reforço de valência que estejam insuficientemente cobertas na equipa.

Na Figura 8 mostra-se o detalhe EDT construído no âmbito do projeto do AIF. Assenta numa hierarquia simples de elementos que decompõem o plano de projeto em tarefas individuais de trabalho, com no máximo 2 ou 3 níveis. A utilização regular do EDT por parte da equipa (semanalmente) constatar o estado da execução do projeto, planear os recursos a usar para entregar os entregáveis nas datas expetáveis. Além disso também ajudou a equipa a definir atividades que seriam necessário executar para criar os entregáveis que se comprometeu concluir.

EDT AIF										
Tarefa ID	Área	Descrição	Horas	Dias	% Concluído	Estado	Responsável	Horas por recurso	Tempo gasto	ETC
1.3	Implementação	1º Módulo - Menus, AIF	2899	362,43	100,00%	OK		2475		0
1.3.1	Implementação	Portal	2051	256,38	100,00%	OK		2061,0		0
	A	Geral	204	25,50	100,00%	OK		244,0		0
1.3.1.1		Inicialização de projecto, preparação de ambientes e estratégia desenvolvimento	40	5,00	100,00%	OK	AA	40,0	40	0
1.3.1.2		Perfis - LDAP	40	5,00	100,00%	OK	PG	40,0	36	0
1.3.1.3		Menu lateral (HTML/CSS/JS)	30	3,75	100,00%	OK	AC	30,0	24	0
1.3.1.4		Navegação de topo (HTML/CSS/JS)	30	3,75	100,00%	OK	AC	30,0	24	0
1.3.1.5		Desenho do modelo de dados (BD)	28	3,50	100,00%	OK	PG	28,0	28,0	0
1.3.1.6		Implementação do modelo de dados (BD)	16	2,00	100,00%	OK	PG	16,0	16,0	0
1.3.1.7.1		Portal - Desenvolvimento da solução: Tratamento de Excepções	20	2,50	100,00%	OK	AA	20,0	0,0	0
1.3.1.7.2		Portal - Desenvolvimento da solução: Mensagens de erro e de sucesso	20	2,50	100,00%	OK	AA	20,0	0,0	0
1.3.1.7.3		Portal - Imprimir ecrãs	40	5,00	100,00%	OK	AA&AC&TA	20,0	20,0	0
	B	PIF e Transição para o AIF	70	8,75	100,00%	OK		70,0		0
1.3.1.8		Desenvolvimento Backing Bean (Java)	16	2,00	100,00%	OK	AA	16,0	16	0
1.3.1.8.1		Inclusão do HTML/CSS/JS na plataforma aplicacional do portal	16	2,00	100,00%	OK	AA	16,0	16	0
1.3.1.9		Desenvolvimento Web Page (HTML, CSS, JS)	10	1,25	100,00%	OK	AC	10,0	6	0
1.3.1.10		Webservices para consulta (SOA)	16	2,00	100,00%	OK	BR	16,0	15	0
1.3.1.11		Queries e procedimentos para consulta de informação (BD)	12	1,50	100,00%	OK	PG	12,0	8	0
	C	AIF - Dashboard Inicial	198	24,75	100,00%	OK		198,0		0
1.3.1.12		Desenvolvimento Backing Bean (Java)	40	5,00	100,00%	OK	BG	40,0	1	0
1.3.1.13		Inclusão do HTML/CSS/JS na plataforma aplicacional do portal	40	5,00	100,00%	OK	BG	40,0	24 (AA) & 80 (BG)	0
1.3.1.14		Desenvolvimento Web Page (HTML, CSS, JS)	70	8,75	100,00%	OK	AC	70,0	32	0
1.3.1.15		Webservices para consulta (SOA)	8	1,00	100,00%	OK	PG	8,0	8	0
1.3.1.16		Queries e procedimentos para consulta de informação (BD)	40	5,00	100,00%	OK	PG	40,0	36	0

Figura 8: Detalhe do documento com a Estrutura de Decomposição de Trabalho do projeto em estudo

Uma vez que a data final estava fixada, teve uma única forma de intervenção, no que à execução do projeto respeita, que foi negociar do conjunto de funcionalidades a incluir e que são possíveis de se implementar dentro do calendário fixado, dando prioridade a funcionalidade com mais baixo custo mas de importante valor para o cliente.

3.1.5.2 *Monitorização do progresso do projeto*

3.1.5.2.1 Atualização do EDT

A atualização do EDT era assegurada pelos próprios elementos da equipa, geralmente à sexta-feira ao fim do dia, para que as percentagens surgissem atualizadas na reunião semanal realizada à segunda-feira a meio do dia.

3.1.5.2.2 Relatórios de progresso semanal ou quinzenal

No projeto AIF, os relatórios em questão eram produzidos semanalmente com os dados enviados por todos os elementos da equipa, sem exceção e, quinzenalmente, ocorria a partilha com a gestão. Os documentos produzidos integravam informação alargada às:

- Atividades executadas no período a que se refere o relatório;
- Atividades que deverão ser executados no espaço de tempo decorrido até ao documento seguinte;
- Aos principais problemas encontrados no período, incluindo alusão a problemas que não tenham sido resolvidos e que aparecerão em relatório para serem solucionados.

Era da responsabilidade do gestor de projeto do AIF receber todas as semanas *inputs* com informação para redigir o relatório e fazer a compilação quinzenal com a gestão do lado do cliente e as restantes partes interessadas.

3.1.5.2.3 Comité de *steering*

Para manter a gestão de topo atualizada sobre o progresso dos trabalhos do projeto, ocorreram mensalmente reuniões do comité *steering*. O objetivo principal desse organismo consistiu, em primeiro lugar, no estabelecimento das bases do negócio (entre as principais partes interessadas), gestão de projetos e do comité do TI com responsabilidade de aprovação do projeto. O comité de *Steering* foi igualmente responsável, em conjunto com os outros gestores (área de negócio, gestores de projeto e TI) pela definição do âmbito geral, do esforço, do custo, do cronograma e desempenho geral do sistema.

3.1.6 Análise e desenho funcional

À medida que a complexidade de um projeto aumenta diminui a probabilidade de corresponder aos requisitos (Wysocki, 2006). No projeto do caso em estudo foram desenvolvidos 2047 requisitos. Antes, porém, de se chegar a esse número final foram realizadas quatro versões do documento de análise funcional (a Tabela 2 contém informação detalhada sobre as entregas) e conduzidos 15 *workshops*. (cinco em cada semana, sendo que a equipa da ConsultIT esteve presente por três vezes e semanas, junto do cliente para inventariar os requisitos da solução).

Uma vez que a comunicação frente-a-frente enfrentava o condicionamento do afastamento geográfico, optou-se por outras formas de comunicação, recorrendo-se à videoconferência, chamadas telefónicas, troca de *emails* e de diversos documentos escritos.

Tabela 2: Entregas da documentação de análise funcional referentes ao AIF

Versão do Documento	Data de entrega
Análise Funcional v1.0	14/03/2016
Análise Funcional v2.0	26/04/2016
Análise Funcional v3.0	31/05/2016
Análise Funcional v4.0	26/06/2016

Tais formas de comunicação alternativa nem sempre se revelaram eficazes, pois ao trocar-se uma conversação cara a cara por uma videoconferência retira-se o nível de proximidade física de conversação. Também se perde o contexto, tornando-se mais difícil negociar, resolver problemas. E quando se passa à conversação via telefone as limitações de comunicação agravam-se, pois perdem-se os gestos visuais, as expressões e as âncoras visuais como apresentações. No caso das comunicações por *email*, as penalizações agravam-se ainda mais, perdendo-se toda a inflexão vocal, o *timing*, e a noção de quais são as ideias que são mais importantes, ficando muito limitado o sentido de um diálogo privado do imediato inerente a conversações verdadeiramente interativas. No caso das comunicações realizadas com recurso a documentos perde-se toda a interação, devendo o autor supor quais os tópicos que interessam e qual o nível de detalhe apropriado. Este meio exclui possibilidade de questões, não sendo suposto haver feedback.

No contexto do projeto e tendo em conta as limitações impostas pelos diversos tipos de contactos com o cliente destinados a estabelecer requisitos, realizaram-se 10-15 sessões de workshops que decorreram nas instalações do BNI, esclarecimentos via *email* e comunicação de novos requisitos e novas funcionalidade por documentos, que confirmando o que se viu anteriormente, se revelam a forma mais difícil de comunicação. Toda a restante troca de informação processava-se por *email* e/ou remessa de documentos escritos.

A matriz de rastreabilidade de requisitos - tabela que liga cada requisito à respetiva origem e segue o seu rasto ao longo ciclo de vida do projeto – apesar de ser uma técnica bastante aconselhada e de trazer importantes benefícios ao negócio e ao projeto, garantindo nomeadamente que todos os requisitos acrescentam valor e ajuda nas suas alterações, optou-se pela sua não aplicação a este caso por consumir muito tempo à equipa.

A identificação dos requisitos apesar de ter sido considerada de início uma atividade de simples confirmação do âmbito, revelou-se bem mais laboriosa. De acordo com o plano inicial, a 5 de maio de 2016 a equipa já deveria ter executado dois meses e meio de desenvolvimento, mas o responsável do projeto (da parte do cliente) continuava a afirmar que o protótipo permanecia em avaliação inicial, não existindo certezas de como deveria funcionar o AIF Evolução. Alegando estar-se ainda numa fase muito experimental, o responsável em questão recusava partilhar os documentos base para

ajudar a construir os gráficos dos indicadores e os modelos de relatórios de supervisão. O gestor de projeto tentou, porém, dissuadi-lo, considerando que apesar que não haver versões finais a equipa precisava dos rascunhos para poder basear o seu desenvolvimento e continuar a dar passos em frente.

Porque todos os outros requisitos não considerados o núcleo do novo sistema e, por isso, nunca incluídos na *baseline* permaneciam sem priorização, estando em causa o seu desenvolvimento, a ConsultIT decidiu proceder à compilação de tudo o que havia sido inventariado até então. No termo desse trabalho, o gestor de projeto promoveu uma reunião de priorização de todos os requisitos aplicando a técnica MoSCoW⁹, a qual consiste em classificar cada um dos requisitos como Must Have, Should Have, Could Have e Wouldn't Have, apurando-se assim as reais prioridades.

Depois dessa reunião de priorização, ocorrida a 26 de maio de 2016 e que juntou as equipas do cliente e da ConsultIT, muitos dos requisitos até então considerados fundamentais desceram de prioridade ou foram excluídos, podendo a equipa, a partir daí, focar-se verdadeiramente no que era importante para este sistema.¹⁰

Ainda a 23 de maio seriam disponibilizados os *templates* para geração dos relatórios de supervisão, o propósito base do AIF, mas só a 8 de Julho de 2016 é que a equipa do cliente conseguiu validar todos os gráficos de indicadores que deveriam ser incluídos e analisados nos relatórios produzidos pela ferramenta agora integrada. Novas dúvidas sobre o projeto seriam colocadas no dia seguinte via e-mail ao cliente, que sobre elas apenas se pronunciaria a 8 de Agosto de 2016, altura em que, no plano original, já deveriam estar a decorrer os testes de integração.

De modo a entender melhor como a tarefa descrita anteriormente foi executada foi calculado o KPI *On-time*. Este indicador de desempenho mede a pontualidade na entrega de um pedido/serviço ou da finalização de uma tarefa de acordo com o plano, ou seja, uma data previamente acordada entre as partes interessadas. Este indicador é

⁹Técnica que caracteriza os requisitos quanto à sua prioridade. O '*Must*' indica que o requisito deve ser obrigatoriamente desenvolvido porque é importante e necessário. '*Should*' indica que o requisito é opcional e é importantes, mas não necessário. '*Could*' indica que o requisito é opcional, desejável mas não necessário. Finalmente a classificação '*Won't*' indica que o requisito não deve ser considerado.

¹⁰ Além dos quatro documentos de análise funcional produzidos e entregues foi ainda produzido um documento que continha todos os pedidos que não constavam na *baseline* do projeto. Este documento nunca chegou a ser entregue uma vez que a reunião de priorização dos requisitos colocou a maior parte dos pedidos prévios com baixa prioridade pelo que não se justificou um documento à parte.

especialmente sensível a atrasos na data de entrega e à disparidade entre a duração prevista para uma tarefa e a duração real. Na análise deste caso irá indicar-se o valor do KPI para cada uma das fases sequenciais que caracterizam o plano de projeto deste estudo.

De seguida mostra-se o cálculo do valor do KPI *On-time* para a referida atividade de análise e desenho funcional. Para o cálculo deste valor utilizou-se a data de entrega do primeiro documento, isto é, a versão 1.

<i>ON-TIME:</i> (Indicador operacional para a atividade análise e desenho)	30,23%
---	--------

O máximo de desvio tolerável para este indicador situa-se aquém dos 20%, sendo que 80% dos projetos de IT apresentam valores inferiores a tal limite. Conclui-se, por isso, que a atividade de desenho e análise foi completada com ligeiro atraso em relação à média considerada *On-time*. Apesar da data da entrega do primeiro documento de análise não ser a data em que a tarefa foi dada como finalizada, pois, como já se viu, houve necessidade de entregar várias outras versões do documento entre outras atividades, pode-se concluir daqui que, já numa das suas fases iniciais, o projeto se revelava atrasado em relação ao plano.

3.1.7 Desenho técnico e desenvolvimento

Rapidamente se percebeu que tecnologias que deveriam ser utilizadas de forma a integrar com facilidade a nova aplicação nos sistemas preexistentes no BNI. Porém, tinham já decorrido algumas semanas de desenvolvimento e não se conseguia dizer ao certo qual seria a melhor forma de apresentar a grande quantidade de informação em numerosos gráficos de indicadores repletos de requisitos específicos. Além disso, não era claro à partida como se iria extrair do sistema a implementar os relatórios de supervisão que deveriam conter toda a informação gráfica que o utilizador decidisse incluir nos documentos. A escolha da tecnologia revelou-se difícil porque uma má opção poderia inviabilizar alguns dos requisitos ou tornar o sistema com baixa

performance, limitação de que padecia a aplicação anterior que se pretendia substituir. A equipa de desenvolvimento viu-se na contingência de efetuar trabalhos de investigação, dada a inexistência de um protótipo a funcionar à data do início do projeto com estas questões de arquitetura bem fechadas.

Neste quadro, o desenvolvimento do AIF acabaria por contemplar o uso das tecnologias para o *frontend*: Java v1.7, Oracle ADF v1.6.0.45 (JSF1.2), AJAX / JSON, HTML5 / CSS3 JavaServer Faces e JavaServer Pages jQuery (correm num servidor aplicacional Oracle Weblogic 10.3).

Para a camada lógica, o serviço de orquestração (SOA) que organiza e trabalha o fluxo de informação entre a base de dados e o Portal AIF, foi utilizado o Oracle Soa Suite v11.1.1.9. A informação foi armazenada numa base de dados Oracle 11g v11.2.0.

No que se refere a armazenamento do repositório de utilizadores recorreu-se ao Oracle Internet Directory (OID) com tecnologia LDAP.

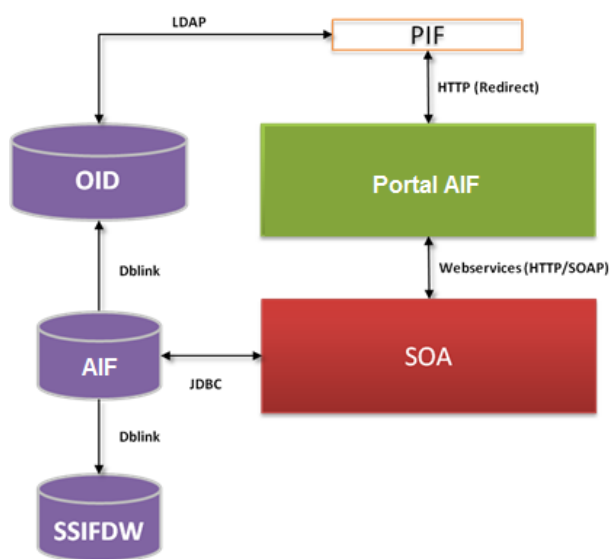


Figura 9: Diagrama do fluxo que descreve o fluxo de informação entre as várias camadas e aplicações.

Tal como refletido no cronograma do projeto (real), a fase de desenvolvimento decorreu durante 22 semanas, repartidas por duas fases (demorando no total 8,5 meses), tendo o BNI expresso expectativa de ter a solução já pronta a testar por volta de 1 de Setembro de 2016, que não se concretizaria, devido aos atrasos sucessivos. Em função desses atrasos, chegou-se a um acordo mediante o qual por volta dessa data apenas

metade das funcionalidades seriam demonstradas, seguindo para testes de aceitação, enquanto que parte da equipa continuariam a avançar com os desenvolvimentos.

ON-TIME: (Indicador operacional para a atividade de desenvolvimento)	116,7%
---	--------

Devido à complexidade do desenvolvimento do *front-end* e consequência também de pouca experiência de alguns dos membros da equipa, essa componente era a que mais influenciava negativamente o projeto, atrasando a respetiva implementação. Consciente disso, o gestor de projeto reforçou a equipa com membros detentores das capacidades requeridas, o que se traduziu numa aceleração da entrega da componente que provocava estrangulamento.

No decurso do período de desenvolvimento também se constatou que alguns requisitos não estavam completos, notando-se, por exemplo, a ausência de detalhe sobre algumas mensagens de erro. A falta de especificação dos requisitos traduz-se no atraso de execução de muitas tarefas de desenvolvimento visto que os programadores não conseguem dedicar-se em exclusivo à produção do código, interrompendo tal trabalho para tentar perceber o que é razoavelmente aceite dentro do âmbito do requisito mal especificado.

À data em que começaram os testes integrados do projeto em estudo, a equipa de desenvolvimento apenas tinha conseguido desenvolver completamente duas das funcionalidades do aplicativo AIF. Além disso, apesar de entregues com atraso em relação ao calendário previsto, essas funcionalidades só foram disponibilizadas devido ao trabalho árduo da equipa em horas extra e fins-de-semana necessário à minimização dos impactos negativos de demoras imputáveis ao cliente.

3.1.8 Testes e passagem a produção

A qualidade do *software* produzido deve ser da responsabilidade de toda a equipa que, ao longo do ciclo de vida do projeto, executa numerosas ações para prevenir

e encontrar defeitos (Stellman e Greene, 2005). Em engenharia de *software* o termo qualidade significa “em conformidade com os requisitos” e quando o software não se comporta como esperado diz-se que tem um defeito.

3.1.8.1 Plano de testes e casos de teste

O objetivo do estabelecimento de um plano de testes para este projeto foi determinar uma lista de tarefas, que, a serem executadas, iriam permitir identificar se todos os requisitos do produto são satisfeitos.

Além da execução do plano de testes e o seu cumprimento também foram detalhados os casos de teste, tarefa esta que requer mais esforço dentro do planeamento das atividades em qualidade. Um caso de teste deve descrever uma interação específica que o *tester* irá efetuar de forma a validar um único comportamento do sistema, no entanto para o caso específico deste projeto, foi incluído mais do um comportamento a verificar num único caso de teste para não alongar em demasia a tarefa de detalhe e execução dos mesmos.

Depois dos programadores entregarem uma *Build* (o primeiro bloco funcional de software), cuja funcionalidade se esperava completa, os *testers* começam a executar o plano de testes referente à funcionalidade disponibilizada. Depois de cada iteração, os *testers* criavam um relatório de testes, documento que contém a lista de todos casos de teste executados e a lista dos que não passaram com sucesso, pretendendo-se tornar visível perante a equipa qual a parte da aplicação que já foi testada e o seu estado. Por cada caso de teste que falhe o *tester* deve criar um relatório de defeito no repositório como o de um sistema de rastreabilidade de defeitos, como foi o caso, HP ALM QC (*HP Application Lifecycle Management Quality Center*) que permitiu guardar a informação sobre os defeitos e encaminhá-los para os *testers*, programadores ou qualquer outro interveniente que, segundo o fluxo de trabalho, devesse contribuir para garantir que o defeito é verificado e reparado.

Finda a fase de testes de integração, seguiu-se a dos testes de aceitação. Tratou-se da última etapa do processo de teste, ocorrendo quando os técnicos de supervisão e chefes de setor e de divisão da aplicação garantiram que conseguiam executar as tarefas

do seu quotidiano e de acordo com as especificações. Esta é uma fase decisiva e crítica e deve preceder sempre a entrada de um novo *software* em produção.

3.1.8.2 Métricas de qualidade do projeto

A qualidade do *software* afigura-se de difícil avaliação, apenas podendo apurá-la de forma indireta pela resposta que garante. Assim, as métricas de qualidade pretendem indicar o nível de resposta do *software* às exigências implícitas ou explícitas do cliente, previamente definidas na especificação dos requisitos.

O quadro seguinte resume, com alguns dados e métricas, o resultado da execução das tarefas de qualidade efetuados no presente caso de estudo:

Funcionalidade	Defeitos detetados por fase				Nº de Requisitos	Nº de Casos de teste	Nº de defeitos/ Requisito	Nº de casos de testes/ defeitos
	Integração	Aceitação	Produção	Total				
Login, acessos e geral	4	0	1	5	10	3	0,50	0,59
Ecrã de transição	3	0	0	3	5	2	0,60	0,66
Menus	4	2	0	6	13	3	0,46	0,5
Dashboard Inicial	8	2	1	11	28	10	0,39	0,9
Alertas - Geral	18	2	0	20	37	5	0,54	0,25
Alertas - Detalhe	33	4	0	37	120	49	0,31	1,31
Ficha Institucional	23	6	1	30	37	8	0,81	0,26
Planeamento	16	3	0	19	40	48	0,48	2,5
Dashboard AIF	11	1	0	12	100	21	0,12	1,75
Detalhe AIF	61	20	2	83	731	131	0,11	1,58
Comentários Gerais	18	5	0	23	248	17	0,09	0,74
Doc. Supervisão	21	0	5	26	425	13	0,06	0,5
Doc. Relevantes	25	2	1	28	50	23	0,56	0,81
<i>Backoffice</i>	103	4	0	107	215	128	0,50	1,2

De notar que a disparidade existente entre o número de casos de teste e o número de requisitos prende-se com o fato de todos os casos de teste terem sido escritos contendo mais do que um comportamento ou requisito, como referido anteriormente.

Poupou-se assim tempo, pois caso contrário seria necessário criar milhares de testes manualmente na ferramenta HP ALM escolhida para o efeito.



Nos testes de aceitação do AIF verificou-se que os utilizadores reais da aplicação tinham a expectativa de que o sistema deveria ter evoluído seguindo o mesmo percursos da evolução do seu modo de trabalho, mesmo que isso não constasse nos requisitos. Os utilizadores esperavam que a aplicação ajudasse o seu trabalho em todas as vertentes mesmo que não tivessem referido certas necessidades por altura do levantamento de requisitos. Um exemplo prático deste problema típico pode verificar-se quando os supervisores constataram que teriam de fazer uma análise dos dados do último trimestre da sua instituição e que, para tal, era necessário compará-los com os do trimestre homólogo do ano anterior. Como nos requisitos e nas discussões sobre o sistema sempre referiram que apenas pretendiam ver os dados dos últimos 13 meses, viam-se privados dos meios para analisar a informação pretendida. Face à insuficiência da nova ferramenta, cuja origem lhe é imputável, sublinham: “É assim que nós estamos a fazer a análise neste momento!”.

De seguida apresenta-se os KPIs On-time para as atividades de testes de integração e testes de aceitação. Como se poder ver, o valor para os testes de aceitação é muito elevado fazendo ressaltar o que se passou na realidade, a tarefa foi apenas iniciada bem depois da data expetável.

ON-TIME: (Indicador operacional para a atividade de testes de integração, apenas 50% das funcionalidades incluídas)	55,0%
ON-TIME: (Indicador operacional para a atividade de testes de aceitação, apenas 50% das funcionalidade incluídas)	63,6%

3.1.8.3 Passagem a produção

A passagem a produção ocorreu muito mais tarde do que se havia planeado, pois só em dezembro de 2016 é que o AIF já tinha sido testado por todos os utilizadores, sendo então consensual o *sign-off* do documento para faze-lo passar à fase final. Sendo o AIF uma aplicação que dependia do novo plano contabilístico que estava a ser adotado pelo BNI a sua passagem a produção só avançaria em maio de 2017. Isto porque, em dezembro de 2016 a aplicação de conversão das contas do antigo plano para o novo plano contabilístico IAS/IFRS evidenciava ainda algumas lacunas e nem todos concordavam com o mapeamento, aguardando-se o estabelecimento de consenso. Sem o mapeamento em produção o AIF não poderia ser utilizado porque necessitava, para construir as suas variáveis e indicadores, das novas contas do plano IAS. Neste contexto, o DSI pensou colocar o AIF em produção só depois de entender que a aplicação de conversão estava a funcionar devidamente, o que apenas conseguiu garantir já depois de Fevereiro de 2017.

Outros problemas contratuais e de disponibilidade de recursos arrastariam a passagem a produção para Maio de 2017, tendo sido encontrados nessa altura os seguintes problemas:

- Discussões entre o cliente e a equipa de desenvolvimento no que diz respeito à interpretação de alguns requisitos e tentativas de incluir mudanças à última da hora. Tais divergências deverão ter que ver com o facto de que os responsáveis agora envolvidos nas questões do AIF já não serem as mesmas que estiveram relacionadas com na sua conceção.
- Dificuldade em manter a equipa de desenvolvimento alocada ao projeto e se garantir a sua necessária motivação. Todos os seus elementos tinham sido,

entretanto, realocados a outros projetos, vendo-se obrigados a voltar a reunir, com entusiasmo reduzido, de forma a resolver questões ainda em aberto.

ON-TIME (Indicador operacional de passagem a produção) ¹¹	151,9%
---	--------

3.1.9 Gestão de riscos do projeto

Risco significa incerteza acerca de uma dada ocorrência que poderá ter um impacto negativo apenas controlável através de uma gestão eficaz. É por isso que no desenvolvimento de qualquer projeto importa ter em conta a implementação de estratégias para tentar evitar perdas, ou provocar a diminuição da frequência das perdas ou da sua severidade.

Na próxima seção procede-se a uma análise qualitativa dos riscos surgidos no decurso de todo o projeto de desenvolvimento do AIF, apontando-se complementarmente as questões desta natureza levantadas aquando da avaliação inicial dos riscos.

3.1.9.1 Riscos do projeto

Desde cedo se apontou como risco do projeto os atrasos sucessivos no *sign-off* dos documentos de análise funcional das fases iniciais respetivas. O processo de aprovação de um documento prolongava-se porque as equipas estavam geograficamente afastadas, limitadas, por isso, na capacidade de assistência e fornecimento dos esclarecimentos necessários. Adicionalmente, as equipas do cliente continuavam a redefinir os requisitos da ferramenta piloto levando muito tempo a comunicar quais deles deveriam constar na solução final. Tais dificuldades agravavam-se pelo fato dos documentos respeitantes ao processo terem sempre de ser aprovados pelos responsáveis dos dois departamentos envolvidos, o DSI e o DTI.

¹¹ No cálculo deste indicador foi considerada como data da definição real dos requisitos a 26/05/2016 quando foi feita a priorização total dos requisitos levantados.

Para aprimorar a ferramenta piloto, realizavam-se trabalhos constantes de atualização de requisitos por parte do cliente, cujas notificações para alterações surgiam tardiamente em resultado da escassa frequência das comunicações entre o cliente e a equipa de desenvolvimento em questão.

De igual modo, identificou-se como risco no projeto do AIF, o desenvolvimento de adaptação dos sistemas de informação ao novo plano contabilístico. Tal risco acabaria por se tornar real, confirmando-se o expectável, traduziu-se na ocorrência de alguns atrasos, uma vez que a equipa que tratava da alteração ao plano não conseguia entregar alguns desenvolvimentos a tempo e testes que ocorriam em ambiente de qualidade afetavam negativamente os testes de integração do AIF, tratando-se, como se tratava, de um ambiente partilhado.

Outro risco identificado no contexto do projeto deste caso de estudo consistia na probabilidade da existência de descoordenação relativamente aos pedidos de alteração com desenvolvimentos da manutenção evolutiva, cuja responsabilidade competia ao DTI do BNI. Devido à deficiente comunicação, sobressaia o risco provável de que as outras áreas de negócio pedissem alterações aos indicadores de risco que estavam a ser trabalhados pelas equipas de desenvolvimento do AIF e, paralelamente, pela equipa de adaptação do plano contabilístico. Este risco acabou por não se concretizar, pois, apesar de ter havido pedidos de alteração, estes não foram efetuados até que os desenvolvimentos fossem dados por concluídos.

Ainda no que diz respeito à adoção do IAS/IFRS se refere e em consequência da escassa frequência de contactos entre os departamentos de supervisão e normas (DSI e DRO, respetivamente) enfrentava-se o risco de que a alteração das contas do antigo plano afetar os indicadores de supervisão sem conhecimento dos técnicos, o que os levaria a basearem-se no antigo plano para escrever os requisitos do aplicativo. Esse risco tornou-se real quando se percebeu que as instituições financeiras iriam deixar de reportar algumas contas do balancete, perdendo-se indicadores importantes, sem que houvesse outros que os substituíssem no novo plano uma vez que esse trabalho não tinha sido pensado.

A meio do projeto, os utilizadores começaram a defender a ideia que só fazia sentido seguir em frente com o projeto e este só iria trazer valor real se fosse incluído também um *workflow* em BPM para aprovações de avaliação dos técnicos às IFs. Porém, essa funcionalidade, por ser considerada um adicional não constava na proposta inicial e apesar da ameaça a sua falta acabaria por não afetar a viabilidade do projeto.

Finalmente, esteve sempre presente como risco ao longo de todo o projeto o constante constrangimento de os consultores externos que apoiavam o cliente nas suas decisões serem substituídos ou retirados a meio do processo, o que a ocorrer se traduziria na perda de muita expertise acerca do negócio, com prejuízo para o apoio às atividades de desenvolvimento. Este risco acabaria por concretizar-se, embora o seu impacto nas atividades de desenvolvimento não tenha sido muito gravoso.

3.1.10 Considerações finais sobre o caso A

Dois pressupostos foram erradamente assumidos neste projeto: 1.º) que o seu âmbito poderia ter sido completamente definido; e o 2.º) que a definição deste âmbito poderia acontecer antes de se dar início a qualquer atividade do projeto.

A falta de sobreposição das fases e o afastamento geográfico das equipas limitou a comunicação entre as partes interessadas. Após as primeiras sessões, a equipa raramente teve a possibilidade de discutir os requisitos com o cliente e informações importantes foram trocadas através de documentos perdendo-se o sentido de relevância da mensagem.

O custo do projeto e a sua duração foram estimados ainda antes de se conhecer os elementos que iriam constituir a respetiva equipa. Além disso, pressupôs-se erradamente que o âmbito do projeto estava bem definido porque o trabalho consistia em replicar apenas uma ferramenta já existente. No entanto, desde cedo se percebeu que devido à fase experimental em que a ferramenta se encontrava, o cliente dificilmente conseguiria garantir que os requisitos que estavam sobre experimentação corresponderiam aos que desejava ter no aplicativo final.

Tornou-se evidente que teria sido necessário um maior envolvimento entre o cliente e o fornecedor, pois as respostas às questões remetidas via *email* ou as aprovações de documentação revelaram-se insuficientes para esclarecer e antecipar dúvidas e soluções.

Em consequência da acumulação de elevada carga de trabalho para os programadores de *front-end* deparou-se a necessidade de incluir novos elementos na equipa. Verificou-se, porém, que nas primeiras semanas de integração dos novos elementos, e visto que o projeto já estava atrasado (segundo o plano), se registou ainda mais desaceleração pois os elementos que já estavam sobrecarregados com tarefas

tiveram que despender parte do seu tempo para elucidar os novos membros da equipa que não estavam ao corrente dos assuntos do projeto nem sabiam bem o que era esperado deles. Ultrapassada a curva de aprendizagem inicial, os três novos elementos conseguiram contribuir positivamente para o avanço do projeto dado possuírem experiência suficiente para serem independentes quando suplantadas as dificuldades iniciais. Convém notar todavia que nem sempre a adição de novos elementos se traduz numa mais-valia, pois os custos iniciais da inserção dos elementos são elevados.

No quadro abaixo listam-se os principais problemas enfrentados pelo projeto em estudo, estimando-se o tempo gasto em cada um. A estimativa baseia-se no cálculo do valor percentual de cada um dos desperdícios identificados tendo por referência a soma das horas realmente executadas por toda a equipa envolvida no projeto (excluindo as férias, feriados e fins de semana). Tal opção permite assegurar a visibilidade do impacto final dos problemas/desperdícios em relação ao todo:

Tabela 3: Identificação dos principais desperdícios do projeto do caso A

ID	Lista dos principais desperdícios	Horas	Nº horas desperdiçadas / Nº horas totais do projeto ¹² (%)
1	Nº de horas a aguardar a definição do âmbito de projeto – funcionalidades prioritárias	1512	8,5
2	Análise e desenho de funcionalidades não prioritárias e que mais tarde foram retiradas do âmbito do projeto	512	2,9
3	Nº de horas a aguardar pela confirmação do cliente de negócio sobre o detalhe de implementação de requisitos prioritários	1696	9,6
4	Alteração do modelo de dados e <i>rework</i> de código por falta de clareza nos requisitos	80	0,5
5	Trabalho em horas extra dos elementos da equipa	276	1,6
6	Incorporação de novos elementos na equipa (<i>developers</i>) em momentos de <i>bottleneck</i> do projeto de forma a não falhar com o prazo de entrega	108	0,6
7	Nº de horas que os <i>testers</i> estiveram a aguardar novas funcionalidades desenvolvidas para executar os testes de integração e nº de horas de indisponibilidade dos ambientes de teste	90	0,5
8	Nº de horas a aguardar o momento adequado para a passagem da aplicação a produção	931,2	5,3
Total		5185,2	29,4

¹² O projeto em análise teve uma duração de execução de aproximadamente 17.700 horas

Como se pode constatar pela tabela 3, o principal e maior desperdício deste projeto surge associado aos longos tempos de espera necessários à conclusão de determinadas atividades, os quais resultaram do atraso nas respostas por partes dos interessados. Outro importante desperdício identificado relaciona-se com o *rework* que teve de ser efetuado em consequência da falta de priorização de requisitos e deficiente comunicação entre os vários *stakeholders* do projeto. Ambos esses fatores contribuíram decisivamente para a redução da produtividade deste projeto tido como tradicional.

As diferentes dificuldades e lacunas identificadas no projeto analisado até este ponto poderiam eventualmente ser suplantadas mediante a aplicação de princípios e ferramentas de uma abordagem de gestão mais Lean e Agile. Nas considerações finais deste trabalho voltar-se-á ao caso A de forma a contrapô-lo com o que são as mais modernas práticas de gestão aplicadas ao IT e especificar em que medida elas poderiam ter ajudado a reduzir quase 30% de horas apuradas como desperdício neste projeto.

3.2 Análise do caso B – Gestão Lean IT de projetos

Para assegurar um melhor desempenho do sistema de aplicações, a gestão da ESI – Espírito Santo Informática (designada NBSI – Novo Banco Sistemas de Informação, a partir de 4 de agosto de 2014¹³ e tem sob sua responsabilidade os serviços partilhados e sistemas de informação do banco) estabeleceu, em 2010, um plano de mudança baseando nos princípios Lean de redução de custos, eliminação de desperdício, standardização, redução do *time-to-market* e ganhos de qualidade.

Antes da decisão de adoção de novas regras na abordagem do sistema informático da instituição, a ESI tinha já desenvolvido esforços de melhoria dos seus processos de desenvolvimento e manutenção aplicacional, com a certificação do CMMI nível 3 e com uma gestão baseada em ITIL e processos de controlo COBIT¹⁴ (Guedes, in *Fibra* 2011). A abordagem Lean surgiria, assim, como o passo seguinte numa caminhada de melhoria de processos, segundo Guedes, P. (2011), administrador da NBSI. A iniciativa do Novo Banco Informática revelar-se-ia como dos casos pioneiros em Portugal de aplicação da Gestão Lean de equipas de desenvolvimento de *software*. Na fase de arranque da implementação do novo sistema nem todos os gestores demonstrariam, todavia, apoio à iniciativa de mudança, temendo complicações, em particular devido ao fato de as suas equipas já estarem completamente sobrecarregadas de trabalho não dispondo de folga para resolução de problemas eventualmente resultantes da adoção da nova abordagem. Mas, ultrapassada a desconfiança inicial, e depois da implementação piloto, o envolvimento alargado no projeto de cerca de 400 ativos distribuídos pelas 30 equipas da NBSI permitiu a obtenção de grandes progressos (Guedes, 2011).

¹³ Resultado da intervenção do Banco de Portugal para garantir os ativos bons do antigo Banco Espírito Santo, empresa mãe da Espírito Santo Informática.

¹⁴ COBIT significa *Control Objectives for Information and Related Technologies*, framework de *governance* de tecnologias de informação.

3.2.1 Enquadramento da organização

A ESI/NBSI dispõe de um quadro de cerca de 550 ativos (incluindo subcontratados) e tem à sua responsabilidade tem cerca de 450 aplicações, 2.300 servidores físicos e 11.000 postos de trabalho. A sua atividade anual consiste em executar 10-15 projetos estratégicos de desenvolvimento e 250 projetos departamentais de desenvolvimento. Ao longo de cada ano implementa, também, em média 700 medidas de manutenção evolutiva e recebe aproximadamente 4. 000 pedidos de equipamento e 20. 000 *tickets* de suporte. A par disso adere anualmente a 1-2 grandes ações de transformação e melhoria interna.

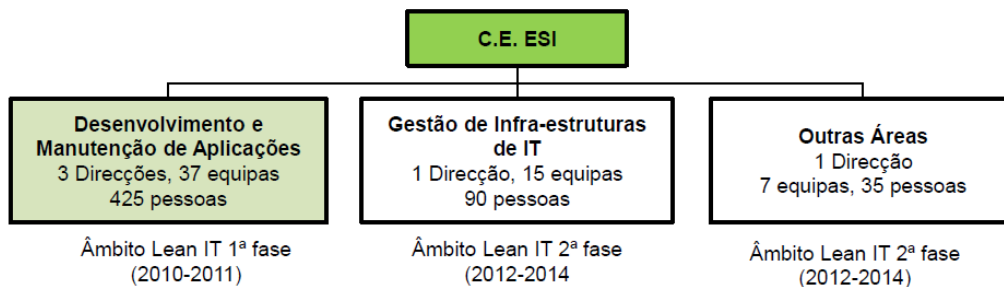


Figura 10: Composição das equipas (Fonte: ESI/NBSI)

A ESI/NBSI decidiu aplicar o Lean IT na gestão das suas equipas tendo em consideração um objetivo prioritário: a melhoria dos processos do IT tidos como essenciais, pois tratava-se (e ainda e ainda se trata) de uma área estratégica e com elevado impacto na atividade e nos custos do banco. A aplicação abrangeu, numa primeira fase, apenas a componente de Desenvolvimento e Manutenção de Aplicações (como se poder ver na Figura 10).

Controlar e melhorar a receção de pedidos de trabalho das áreas de negócio, criar um sistema de gestão de trabalho diário visível, derrubar possíveis divisões existentes entre equipas de trabalho através de uma mudança de cultura assumem-se como outras das razões pelas quais os gestores da equipa de informática do então Grupo BES se decidiram pela adoção de uma outra abordagem, que seria implementada em colaboração com um grupo de consultoria externa da McKinsey, com provas de sucesso na execução de projetos semelhantes no sector bancário.

O processo de mudança arrancou com a realização de reuniões envolvendo várias áreas da empresa escolhidas para a fase piloto. Nesses encontros, em que participavam arquitetos, analistas de negócio, programadores, gestores de produto e líderes, estiveram em debate questões como “O quê?”, “Porquê?” e “Como?”, procedendo-se, também aí, a um levantamento do estado atual e à identificação dos principais limitadores de produtividade.

Em resultado destas primeiras reuniões e trabalhos de diagnóstico às lacunas dos processos no ESI/NBSI fixaram-se como objetivos principais os seguintes tópicos (Fonte: ESI):

- Focar as equipas nas atividades mais prioritárias
- Fornecer indicadores de medida e gestão das atividades da equipa
- Transformar o Coordenador de equipa num bom gestor de equipa
- Envolver todos os elementos da equipa
- Criar uma cultura de gestão centrada na produtividade

3.2.2 Transformação Lean

Como referido anteriormente, a transformação Lean do ESI/NBSI iniciou-se em 2010, mais concretamente em abril de 2010, com a aplicação do primeiro piloto a apenas a três equipas. Ainda nesse ano, o projeto seria gradualmente aplicado a mais três equipas de desenvolvimento, entre outubro e dezembro. Em 2011 e até ao primeiro trimestre de 2012 tornou-se possível fazer o *roll-out*, aplicando-se a iniciativa a um total de 28 equipas (ver Figura 11). A aplicação do Lean IT nas equipas de infraestrutura deu-se mais tarde, entre 2012 e 2013. A aplicação gradual e por vagas do Lean às equipas teve por objetivo garantir o sucesso da abordagem, assegurar a sua sustentabilidade futura e colher os frutos das lições aprendidas internamente.

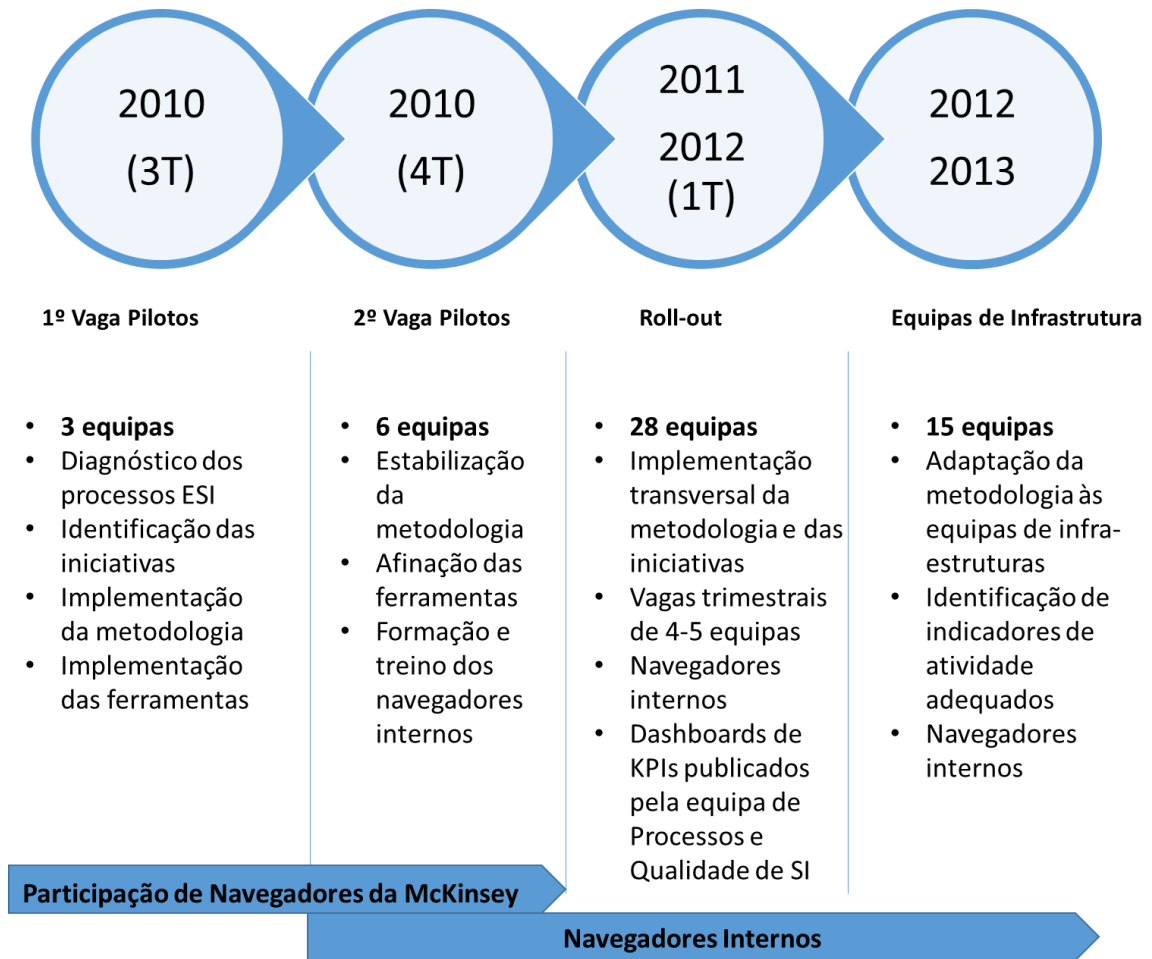


Figura 11: Cronologia da Transformação Lean IT (Fonte: ESI/NBSI)

Transformação em cada equipa

A transformação Lean IT em cada uma das equipas desenvolveu-se por três fases, como se constata na figura seguinte.



Figura 12: Processo de implementação do Lean IT numa equipa (Fonte: ESI/NBSI)

A primeira fase, planeada para decorrer durante uma semana, arrancou com *Workshops* em que era apresentado a todos os membros da equipa o levantamento dos problemas encontrados nos processos da organização bem como o programa Lean como perspectiva de solução. Nesses encontros havia ainda oportunidade para os coordenadores, que já tinham participado no programa Lean, apresentarem testemunhos sobre a execução da abordagem em outras equipas.

A segunda fase consistiu na implementação das iniciativas, sempre em coordenação com a direção, que conhecia o *status* do projeto para que esta pudesse ajudar a ultrapassar eventuais dificuldades entretanto surgidas. Com a duração de cinco semanas esta etapa permitiu a consolidação e melhoria contínua do até então implementado. Também executada em cinco semanas a última fase do processo concluiu-se com uma reunião alargada à Direção e Comissão Executiva, onde se apresentaram os indicadores da equipa e se partilhou as principais dificuldades derivadas da aplicação das novas práticas.

3.2.2.1 Reorganização da equipa

A reorganização do ambiente de trabalho da equipa revela-se fundamental para facilitar as comunicações entre pares e minimizar distrações externas, sendo que uma má comunicação e distrações externas reduzem a produtividade de uma equipa. Ora, antes da transformação Lean, os membros das equipas da ESI/NBSI dispersavam-se por um número excessivo de tarefas, queixavam-se de interrupções frequentes e trocas de projeto geradoras de desperdício, perda de foco e frustração. Frequentemente, os problemas e as atividades de suporte à produção retiravam ativos de projetos relativos a novas implementações, o que impactava negativamente nos clientes, sendo afetada, em consequência, a perceção que estes tinham sobre o IT. Além disso, os membros da equipa de desenvolvimento de *software* tinham todo o tempo alocado, não dispoñdo de oportunidade para se dedicarem com calma à melhoria dos processos internos.

As organizações como a ESI/NBSI procuram investir nas pessoas e colocam as suas expetativas mais elevadas de maneira a torna-las autossuficientes de forma a que consigam resolver os problemas com que se deparam. Antes da adoção da abordagem Lean, os recursos das equipas da ESI/NBSI estava alocados a apenas uma aplicação e

era sobre esta que detinham o conhecimento, desconhecendo em detalhe o funcionamento das outras aplicações. Ora isso constitui, segundo o novo entendimento adotado, desperdício por não permitir um aproveitamento em pleno das capacidades dos recursos. Em consequência, decidiu-se reorganizar a equipa em várias *pools*, nas quais cada ativo ficaria alocado a mais do que uma aplicação fazendo com que se diminuísse o desperdício do aproveitamento das suas capacidades. Cada *pool* passaria, então, a revelar uma constituição de elementos mais transversal e não tão rígida, permitindo-se o intercâmbio de elementos entre *pools* caso necessário e garantindo-se à *pool* maior independência, sempre que oportuno, pois esta ganharia uma maior transversalidade. Uma tal estratégia apresenta a vantagem assegurar um planeamento mais adaptativo e ajustado, pois passa a planear-se a pensar nas *pools* e não nos recursos individuais, reduzindo-se também o risco associado.

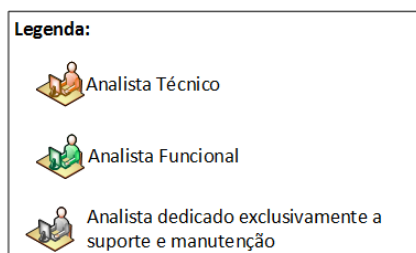


Figura 13: Alocação dos recursos ESI antes da abordagem Lean (Fonte: ESI/NBSI)

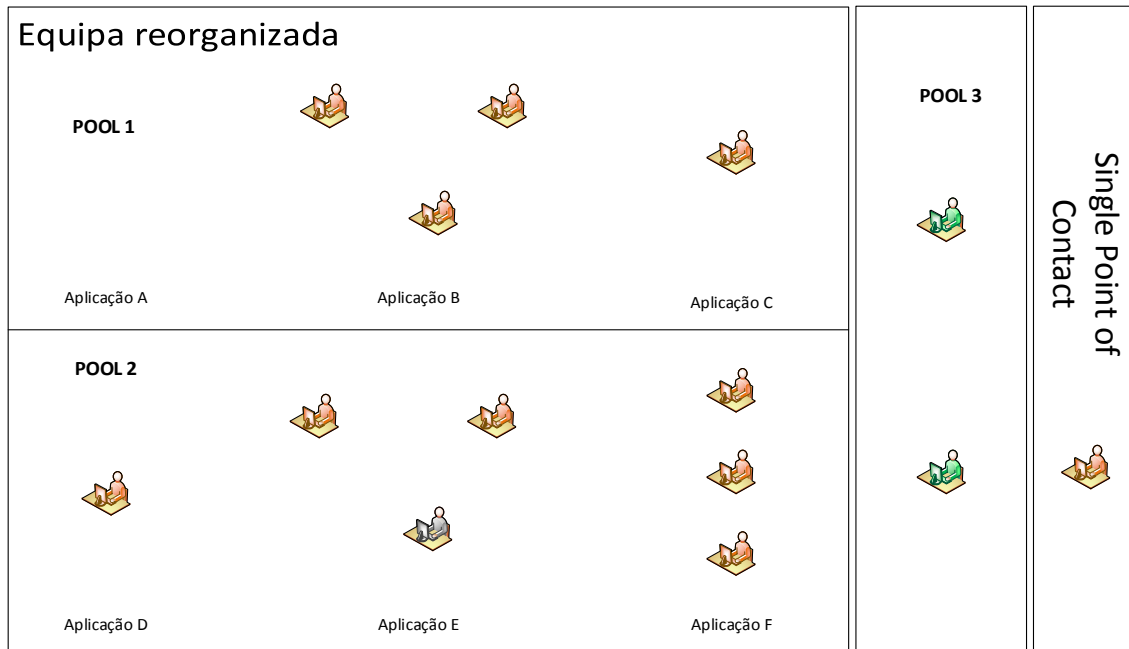


Figura 14: Organização da equipa por pools em consequência da transformação Lean IT (Fonte: ESI/NBSI)

Adicionalmente, em cada *pool* elegia-se um elemento para exercer funções de *Single Point of Contact*¹⁵ (SPC), competindo-lhe o fornecimento, quando necessário, esclarecimentos relativos ao trabalho do grupo. Com uma tal opção evitava-se a possibilidade de interrupção aleatória do trabalho de um ou mais elementos da equipa, em caso de um primeiro contacto não assegurar o acesso escolhido à informação pretendida.

Tendo em conta que, importa estabelecer-se a forma como os elementos das equipas comunicam entre si, porque os problemas encontrados no dia-a-dia assumem uma tal complexidade que impede um único ativo de assegurar a sua resolução e torna o trabalho em equipa uma questão, para promover a eficiência das comunicações e prevenir a interrupção do trabalho procedeu-se à introdução do SPC nas equipas da NBSI, adotando-se um procedimento idêntico ao concretizado pela indústria automóvel através da criação do líder de zona. Aqui o SPC surge investido das responsabilidades acrescidas de facilitar a reunião diária com suporte do quadro de controlo visual e de seguir o fluxo de trabalho de modo a entender os possíveis interruptores do processo. O estabelecimento de tal função garante a concentração num único ponto de foco a

¹⁵ *Single Point of Contact* é atribuído a um elemento da equipa para permitir centralizar e facilitar as comunicações de grupo de maneira a diminuir a entropia entre os elementos das equipas

prestação de contas que impulsiona a mudança proactiva e o seguimento da melhoria contínua. Além do mais, e como porque se assume como característica revelante das equipas de alto desempenho a diversidade interna de opiniões por vezes muito díspares dos ativos que as constituem, impõe-se a necessidade de intervenção de facilitador de conflitos. Tal papel, igualmente atribuído ao SPC, sendo bem desempenhado garante à equipa uma reforçada capacidade de resolução de problemas, assegurando-lhe maior abrangência na abordagem a dificuldades e à respetiva solução.

A reorganização de equipas operada no seio da ESI/NBSI em consequência da aposta na implementação da abordagem Lean IT trouxe como outra consequência positiva, a possibilidade de disponibilização de tempo disponível para os recursos se dedicarem, caso necessário, à execução de tarefas não planeadas. Entre essas tarefas figuram, por exemplo, as relacionadas com a pesquisa da melhor forma de lidar com os incidentes em produção e de garantir que todos eles serão tratados e endereçados com redução de esforço associado e sem envolvimento de vários elementos no tratamento/ leitura e /análise do mesmo problema. O tempo reservado a atividades não planeadas assume a designação, no contexto do caso de estudo, de *buffer*. Face à necessidade das equipas reportarem todas à mesma administração verificou-se por outro lado uma partilha entre elas das descobertas e melhorias que obtinham, observando-se aqui a máxima Lean segundo a qual constitui uma importante mais-valia a amplificação/aprofundamento do conhecimento.

Em suma, a reorganização das equipas, entretanto tornadas Lean na ESI/NBSI, permitiu a simplificação na afetação dos recursos, daí resultando uma maior clareza sobre o desperdício existentes no planeamento, a qual deixaria a descoberto processos que poderiam ser standardizados nas diferentes *pools*, assim como outras ineficiências.

3.2.2.2 Planeamento a curto prazo (semanal)

A agilidade é conseguida se se mantiver os projetos de dimensão reduzida em termos de âmbito e duração, pois as decisões podem ser tomadas rapidamente sem adicionar custo ou risco. Esta base de pensamento constituiu uma das razões pelas quais se decidiu planejar detalhadamente as atividades das equipas da ESI/NBSI, apenas a

curto prazo, tanto no que refere a todas as *pools* o que diz respeito às tarefas dos respetivos elementos.

Assim, em primeiro lugar, do portfólio de projetos com priorização de atividades/tarefas já efetuada e de que são conhecidos os benefícios tangíveis, retira-se o conjunto de trabalho que as equipas devem realizar na semana, procedendo-se, depois, a uma definição e atribuição de tarefas a cada *pool*. O passo seguinte consistirá, por outro lado, na distribuição de tarefas (definidas com granularidade de duas até quatro horas) a executar por cada elemento de cada *pool*. Além disso, ao *buffer* é sempre reservado um tempo destinado à concretização das suas atribuições específicas, sendo, igualmente, previsto no plano semanal um espaço de folga que permitirá adicionar as tarefas não expectáveis sem causar grande quebra no programa. Tal flexibilidade é suportada por planeamento «heijunka», termo proveniente do Lean da indústria, que significa nivelamento de produção e que permite misturar a sequência do trabalho numa fila a qualquer momento sem causar impacto negativo.

Para suportar esta nova forma de distribuição de trabalho e organização de equipas procedeu-se, entretanto, ao desenvolvido um *software* (maioritariamente em Excel) capaz de assegurar o seguimento das atividades e o seu estado, produzir os relatórios, imprimir os artefactos gráficos com KPIs e os cartões para os Quadros de Controlo Visual. Através dessa ferramenta torna-se possível verificar, em cada semana, as horas planeadas para cada recurso e o tempo (em horas) reservado para o *buffer* e o total de horas de toda a equipa.

Esse novo método de planeamento permitiu diminuir a complexidade do agendamento das tarefas dos recursos e facilitou a gestão dos pedidos das áreas de negócio. Veio, de igual modo, traduzir-se na redução do desperdício em termos de horas identificadas como pouco produtivas, isso em resultado direto da análise de ineficiências imputáveis ao não aproveitamento da capacidade máxima dos recursos, possibilitando assegurar um maior equilíbrio na distribuição da carga de trabalho pelos ativos.

O progresso da execução do trabalho em relação ao plano passou a ser seguido através dos gráficos de *burn-up* (ver exemplo na Figura 15) que mostram as horas executadas contra as horas totais planeadas para completar as tarefas. Esses registos são revistos diariamente para se garantir que o progresso está a seguir o ritmo apropriado e

se avaliar se o trabalho foi terminado no dia anterior, estimando-se o que vai ser executado no próprio dia, mediante quantificação do tempo necessário à conclusão das tarefas em curso e discussão de constrangimentos ou barreiras.

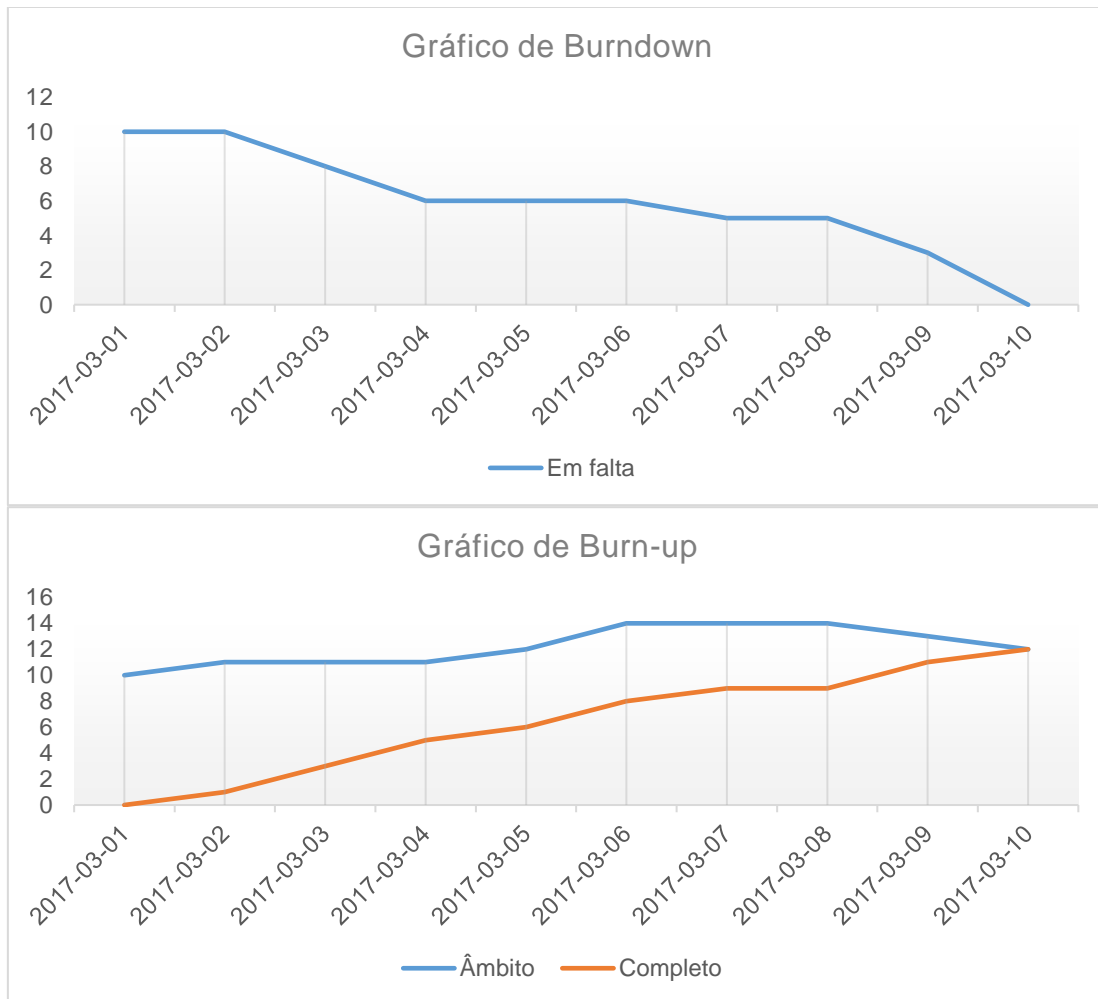


Figura 15:Dois tipos de gráficos muito utilizados para seguir e comunicar o progresso dos projetos, os gráficos de Burndown e de Burn-Up.

O gráfico *burndown* mostra a quantidade remanescente de trabalho a executar enquanto o *burn-up* mostra a quantidade executada e a quantidade de trabalho total prevista. O exemplo da figura mostra os dois tipos de gráfico para o mesmo projeto, podendo concluir-se que o *burn-up* é mais completo garantindo uma amostra geral sobre o âmbito. Neste caso vê-se o aumento do âmbito e depois a sua diminuição próximo da data final. Uma leitura exclusiva do *burndown* poderia induzir no erro de se pensar que a equipa apenas aumentou o ritmo de trabalho na parte final do projeto.

whiteboard, ferramenta a detalhar na próxima seção, tornou possível começar a

No *whiteboard* da ESI/NBSI consta a lista de pools, apresentando cada uma delas uma subdivisão respeitante à equipa, como referido anteriormente no capítulo da reorganização. Os ativos afetos a cada pool surgem discriminados, sendo também referenciadas as tarefas atribuídas. As tarefas são colocadas no quadro em cartões, com ímanes, associadas a cada dia da semana e, quando a tarefa não está fixada em nenhum dos dias da semana, pode estar no estado «DONE», significando que foi terminada, ou «HELP», o que indica que o elemento responsável pela sua execução precisa de ajuda para a conseguir terminar dentro do prazo estipulado. Pode, ainda, encontra-se no estado «PARK», sempre que se encontrar inativa no momento por variadas razões como, por exemplo, a aguardar pedido de informação extra ao stakeholders.

Adicionalmente pode, de igual modo, atribuir-se às tarefas os filtros «HIGH» ou «LOW» para indicar que são consideradas de importância elevado ou baixa.

No quadro de controlo visual da ESI/ NBSI figura também um item designado de *Mood* ou moral do elemento da equipa. Esta seção permite aos elementos do grupo permite de forma fácil os recursos poderem comunicar de forma fácil comunicar entre si o estado das suas tarefas e expressar possíveis preocupações relacionadas à sua execução. Quando assinala o *mood* a verde o elemento da equipa indica que as tarefas em geral serão finalizadas dentro do tempo estipulado, se escolhe o amarelo assinala que as tarefas estão a risco de não acabarem dentro do tempo previsto, circunstância que se exige uma medida de ação corretiva, e quando indica o *mood* vermelho sinaliza que as tarefas não serão concluídas dentro do tempo estimado, impondo-se uma intervenção corretiva. Os membros da equipa estão autorizados a mudar o seu estado da moral, sendo que, com o fornecimento de informações dessa natureza, podem prestar auxílio na medida em que ajudam a equipa atinja os objetivos a que se propôs. Tal tipo de ferramenta permite a abertura dos elementos aos problemas do dia-a-dia e fomenta a autoajuda, uma vez que todos possuem a consciência plena do trabalho que está terminado e do que ainda está por fazer por parte de cada um deles.

No canto inferior direito do quadro surge, por outro lado, um espaço reservado ao registo de problemas, em que é possível indicar os problemas surgidos ao longo do desenvolvimento do trabalho, apontando-se uma possível solução, o responsável pela situação e se sugerem os próximos passos a dar para resolver a questão.

Indicadores do Whiteboard

O quadro de controlo visual cujo principal objetivo consiste em providenciar num relance de olhar em 15 segundos, informação sobre o estado de projeto a quem por ele passe deve ser constituído por alguns KPIs de execução. Para tanto, inclui uma área de monitorização e controlo reservada à amostragem dos indicadores da equipa. Entre vários indicadores destacam-se:

- *Distribuição das horas executadas por tarefas:*
 - Principais
 - Não planeadas
 - Não terminadas
 - Secundárias de baixa prioridade

- *Evolução do burn-up (%)* por cada dia da semana, com uma linha de tendência semanal;
- *Eficiência de evolução das tarefas* (tempo planeado vs. tempo gasto)
- *Trabalho não planeado vs planeado*

Esses indicadores são todos revisitados diariamente pela equipa durante as reuniões matinais, como abaixo se explicita.

3.2.2.4 Reuniões de *whiteboard* diárias

As reuniões diárias implementadas na ESI/NBSI não devem demorar mais do que 15 minutos, no horário que mais convém a cada equipa e procuram juntar todos os seus membros a quem é requerida participação ativa. Aí, cada um dos elementos sumariza as tarefas executadas até ao momento, indicando as que farão até ao dia da reunião seguinte, antecipando problemas para cuja solução vão necessitar de ajuda. Como não se trata de um encontro destinado à resolução de problemas, apenas se procede à sua identificação e, se algum tópico exige uma discussão mais prolongada, agenda-se uma reunião detalhada reservada às partes interessadas.

É também nas reuniões diárias que se analisa o *Whiteboard*, abrangendo essa análise o trabalho executado contra o total planeado, o que constitui um meio eficaz de a equipa se ajudar a ela própria a tomar ações corretivas e preventivas. Ao focar-se no trabalho realizado por cada um dos seus membros e naquilo que irá ser realizado no dia, a equipa assegura uma visão do trabalho em mãos, funcionando a reunião diária não só como via para atualização de estado do progresso das tarefas onde os indicadores acima descritos são revisitados, mas também como meios de comprometimento dos elementos do grupo em relação aquilo será realizado no próprio dia. E daí resultar num verdadeiro espírito de equipa.

3.2.2.5 Planeamento a médio e longo prazo

Planeamento a médio prazo

Em matéria de planeamento, e tal como o previsto na abordagem Lean a ESI/NBSI procede ao estabelecimento de planos a alto nível que têm como base o *Backlog* de projetos que as equipas tenham capacidade de desenvolver num único mês de trabalho. O plano em causa detalha as tarefas expectáveis no âmbito dos projetos que vão entrar em produção, aceitação, em análise de requisitos ou em orçamentação no mês respetivo.

Planeamento a Longo Prazo

A identificação de potenciais projetos constitui responsabilidade dos gestores de negócio, a quem incumbe a tarefa de descrever as necessidades dos seus respetivos departamentos e/ou produtos. Os gestores de negócio dos departamentos priorizavam cada item contra os itens previamente existentes no *Backlog*. Na ESI/NBSI, estabeleceu-se um sistema de aceitação de pedidos de trabalho de prioridade elevada até três meses de capacidade, ou seja, o trabalho planeado é mantido de forma estável com uma previsão trimestral.

Em situações em que a chegada de pedidos de projetos ultrapassa a capacidade da equipa num período de três meses, é criada uma fila que acumula as solicitações

excedentárias em relação ao potencial de resposta. A decisão da administração consiste em acumular esses pedidos, depois de fazer uma avaliação primária do seu possível custo, sem entrar em grande detalhe acerca das especificações. A orçamentação mais detalhada e a análise funcional dos requisitos ficarão a aguardar para quando o projeto for efetivamente planeado. O fluxo de trabalho permanece, não avança para se evitar a acumulação de tarefas não concluídas (inventário do Lean) que representaria um desperdício e possível fonte de defeitos.

Torna-se importante que a acumulação de projetos nessa fila de tarefas sem data prevista (a FIFO) não ultrapasse o *time-to-market* exigido pela estratégia do banco, sendo minimizadas, por via da discussão entre as partes interessadas, nomeadamente entre os diretores dos vários departamentos envolvidos.

Na maioria dos casos de desenvolvimento de software os *backlog* de projetos superam a capacidade das equipas, pelo que, numa abordagem mais sustentável para as equipas da ESI/NBSI teve de ser tomada, em vez de adaptarem a sua velocidade às necessidades desenvolveram uma cadência de desenvolvimento que se perspetivou confortável ao longo do tempo. Daí resultou um nivelamento do ritmo e a estabilização da alocação da capacidade por cada período, aplicando-se aqui mais uma vez o «heijunka».

3.2.2.6 Reuniões de feedback

As reuniões de feedback implementadas pela ESI/NBSI ocorrem trimestralmente e juntam, em encontros separados, o coordenador e cada um dos elementos da equipa. O objetivo de tais reuniões consiste em analisar os KPIs individuais e avaliar, com vista a atuações futuras, as lições aprendidas no trabalho já desenvolvido, as quais devem ser aproveitadas para melhorar o contributo dos ativos no trabalho coletivo.

Para promover o contínuo esforço na conquista da excelência e aprendizagem contínua, o coordenador avalia as tendências de execução da equipa como um todo e analisa o que poderá ser otimizado individualmente, procedendo, de igual modo, à

aferição da moral individual, considerando que os trabalhadores são o ativo mais importante e devem ser tomadas em conta e o seu ponto de vista deve ser respeitado.

3.2.2.7 Aplicação da melhoria contínua

Graças à adoção da abordagem Lean IT, as equipas de desenvolvimento e manutenção da ESI/NBSI passaram a assegurar o levantamento e identificação de medidas de melhoria da sua própria atividade individual ou global da organização.

Sabe-se da literatura que a forma como as organizações tratam os seus incidentes em produção diz muito acerca do seu grau de maturidade. Uma grande organização poderá ter centenas ou mesmo milhares de defeitos não descobertos, sendo isso suficiente para provocar uma reação em cadeia e resultar numa interrupção de elevada severidade (Schmidt & Lyle, 2010). A resolução destes incidentes tem muito a ver com o Lean pois a criação de uma cultura atenta às pequenas falhas representa a base para a prevenção e cada defeito surge associado à falta de conhecimento sobre o sistema e as suas interações. É por isso que a abordagem Lean IT promove a aposta na resolução de pequenos problemas de forma disciplinada para prevenir maiores danos. Seguindo uma tal orientação, a ESI/NBSI cultiva uma cultura de melhoria contínua, a qual surge orientada potenciar a resolução de problemas nas aplicações que geram incidentes em produção. Promove, paralelamente, a simplificação, standardização e otimização de processos de passagem a produção de modo a reduzir a sua duração. Ao garantir que todas essas atividades são executadas da mesma maneira, a ESI/NBSI projeta a redução da variabilidade¹⁶ do trabalho, cumprindo um dos objetivos principais, que se traduz numa diminuição efetiva do lead-time, da frequência e da extensão de possíveis erros.

A sistematização de tarefas diárias com recurso a *checklists* de verificação de jobs noturnos e a sua standardização e automação, além permitirem a redução da

¹⁶ Neste contexto a variabilidade abrange dois tipos: a variabilidade na procura (que pode significar a variação do número de projetos a desenvolver ao longo do ano, a variação nos requisitos de negócio e variação no número de *tickets* recebidos) e variabilidade no desenvolvimento (que pode implicar variação no tempo e qualidade do trabalho de pessoas na mesma equipa bem como a variação nos processos de trabalho de equipas diferentes)

variabilidade implicam uma diminuição do tempo de execução, o que garante à equipa mais tempo para atividades criativas e resolução de problemas.

No quadro da aposta na melhoria contínua a ESI/NBSI procedeu a uma análise detalhada da situação pré-existente à mudança no domínio da comunicação, concluindo pela existência de situações pontuais em que a informação entre as equipas não fluía de forma conveniente. As equipas chegavam a desconhecer que algumas tarefas ou projetos onde participavam diretamente ou em que estavam envolvidas tinham sido canceladas por elementos de outras equipas. Para derrubar barreiras e combater a perda de fluidez de informação observada neste tipo de *handoffs* os responsáveis pelo programa de mudança sugeriram que as equipas trabalhassem a melhoria da comunicação através da implementação de reuniões semanais ou quinzenais entre equipas. Este aperfeiçoamento revela-se importante para os maiores projetos, pois constitui um ponto de partida para eliminar a raiz de possíveis desperdícios nas ações do dia-a-dia dos elementos das equipas Lean.

3.2.2.8 Controlo e monitorização

As métricas de desempenho assumem-se como a base do controlo de processo e melhoria, sendo que, nos termos das metodologias tradicionais, os controlos e as medidas se focam no custo e no cronograma, sofrendo revisões periódicas em grande e complexa escala, de que resultam atrasos nos trabalhos. Pelo contrário, no caso do desenvolvimento Lean, as medidas são mais simples, visuais e imediatas.

Uma vez que a iniciativa Lean e as medidas resultantes da sua aplicação não podem ser aplicadas com ligeireza, a gestão deve estar envolvida de forma comprometida na verificação das métricas e progresso. Um dos grandes desafios que o IT enfrenta consiste, exatamente, na demonstração do valor que adiciona ao negócio, que de certa forma constituirá a evidência do seu valor. A aplicação Lean vem, aliás, ajudar a provar o que certa forma antes era difícil, que o IT não traz apenas custo, pois permite a monitorização dos indicadores do grau de satisfação dos clientes, do número de incidentes em produção ou mesmo dos indicadores de desempenho das equipas.

Na ESI/NBSI o processo é revisto pela administração a cada trimestre, sendo providenciado feedback em resposta à análise dos KPIs e das iniciativas de melhoria. A administração defende que a monitorização dos KPIs assume extrema utilidade para a gestão dos projetos Lean IT, sendo igualmente relevante o desenvolvimento de métricas de para que seja possível verificar, por um lado, o desempenho das equipas na produção de software, e, por outro, se os requisitos do cliente estão a ser tomados em consideração. A medição regular dos indicadores permite ver o desperdício onde antes se apresentava oculto e, desta forma, trabalhar na melhoria dos processos de forma contínua.

Dentro dos numerosos KPIs de gestão existentes, a ESI/ NBSI optou para medição e controlo do desempenho do Lean IT pelos seguintes:

- Burn-up (% da execução de tarefas principais)
- Tarefas principais (% da execução das tarefas principais por total de tarefas planeadas)
- Trabalho não planeado (% dessas tarefas em relação ao total)
- Diferença entre o *buffer* reservado e a realização do trabalho não planeado
- Eficiência de execução (# Horas executadas vs. Horas planeadas para tarefas terminadas)
- Utilização de capacidade (# de Horas previstas vs. Horas trabalhadas)

3.2.3 Considerações finais sobre o caso B

As equipas da ESI/NBSI viveram a transformação Lean no seu quotidiano laboral, adquirindo, em consequência, uma considerável experiência sobre a perspetiva da nova abordagem e do que ela significa em termos de cultura e metodologias de trabalho. No quadro abaixo inscrito pode ver-se a evolução dos indicadores de desempenho aplicados transversalmente a todas as equipas no âmbito da mudança operada pelo Lean:

KPI	Valores Iniciais	Valores em 2013	Valores em 2015
<i>Burn-up</i> (% execução das tarefas principais)	50 - 60%	85 - 90%	89 - 95%
Tarefas Principais (% das tarefas principais nas tarefas planeadas)	85%	85%	90%
Trabalho não planeado (% em relação ao total)	30 - 40%	5 - 10%	2 - 5%
Diferença entre o <i>buffer</i> e o trabalho não planeado	-10 a +10%	< 5%	< 5%
Eficiência de execução (# horas executadas vs planeadas para tarefas completadas)	80 - 120%	~95%	~98%
Utilização da capacidade (% de horas previstas vs trabalhadas)	~100%	~100%	~100%

Pelos dados acima considerados pode concluir-se que a **percentagem da conclusão das tarefas** (principais) aumentou cerca de 35-40%, o que significa que os planos passaram a ser realmente cumpridos, cabendo aos coordenadores e responsabilidade de puxar pela equipa e incentivar o cumprimento dos seus objetivos semanais e, consecutivamente, trimestrais e anuais.

Verificou-se também que a **percentagem de tarefas não consideradas principais** diminuiu ao longo dos anos, o que leva a crer que o trabalho executado respeitou maioritariamente tarefas de prioridade mais elevada. Tal terá resultado da priorização e foco se orientarem para o cliente, combatendo-se dessa forma o desperdício.

Em virtude das numerosas ações de melhoria contínua e da constituição do *Buffer* no planeamento semanal a **percentagem de trabalho não planeado** caiu para 2-5%, um valor muito baixo quando se considerar o ponto de partida, que se começou a implementar a mudança, a aplicar a iniciativa, em que rondava os 30-40%. Antes, a

elevada componente de trabalho não planeado um problema sério para a ESI/NBSI, deixando de ser em resultado da intervenção.

Na época pré-Lean, verificava-se que o tempo destinado ao trabalho não planeado era sempre superior ou inferior ao estimado, daí resultando um constante desacerto entre o tempo real e o reservado no planeamento para a execução de tarefas. Depois da aplicação dos princípios Lean constatou-se ser possível estimar com maior rigor o tempo real do trabalho, tornando-se até possível às próprias equipas ajustar o *buffer* em função dos indicadores semanais, ganhando, com isso, adaptabilidade. Tal mudança traduziu-se numa melhor gestão de tempo e dos recursos, permitindo uma redução da variabilidade no processo, que passou a situar-se em -5 a 5%.

Relativamente ao indicador **Eficiência de Execução** (número de horas planeadas *versus* número de horas planeadas para tarefas completadas) registou-se, de igual modo uma melhoria significativa: antes do Lean as equipas chegavam a executar horas a mais para além do planeado, como se pode depreender do indicador superior a 100% ou se constatava o oposto, que as tarefas eram estimadas por baixo. O ideal para este indicador é ter um valor próximo de 100%, valor que foi o alcançado com a iniciativa, conseguindo-se, na prática planeamentos mais realistas e um sério comprometimento em relação ao plano concebido.

Ao analisar-se o indicador de **Utilização de Capacidade**, que se manteve próximo dos 100%, pode concluir-se que, em resultado do conjunto de medidas e ferramentas introduzidas, a capacidade utilizada manteve-se máxima podendo-se constatar que os recursos continuam a ser utilizados em plenitude e não se verificou desperdício em termos de capacidade.

Além do já foi referido, ainda se pode apurar que em apenas dois anos e meio a ESI/NBSI viu a sua produtividade aumentar em cerca **23%**, quando calculada em termos de custo, e em 27%, se no cálculo tiver por referência as horas despendidas a executar as tarefas¹⁷.

O foco Lean nas necessidades do cliente e a forte orientação aos resultados permitiu a integração das equipas de IT com o resto dos *stakeholders* da organização, assim como a melhoria da sua colaboração ao longo do ciclo de vida do *software*,

¹⁷ Dados referentes ao ano de 2012.

sentindo-se isso em consequência da aplicação direta de ferramentas Lean simples, como o *Whiteboard*, que alavancou a aproximação e a inclusão de todos.

Uma das principais vertentes do Lean em IT consiste em preparar e planear ao ativos, otimizando as suas tarefas individuais para que elas possam ser bem executadas à primeira, criando valor para o cliente e minimizando o desperdício. Neste quadro, a ESI/NBSI conseguiu adaptar a capacidade dos seus recursos ao volume das atividades provenientes da procura, facilitando o ajuste equilibrado do IT às exigências do mercado.

Modificaram-se algumas funções e definições em resposta à necessidade de adoção de uma nova cultura, estandardizaram-se e especializaram-se processos nunca esquecendo que a abordagem a seguir era progressiva e iterativa para permitir a assimilação da mudança e de modo a torná-la sustentável com o tempo. A implementação dos KPIs anteriormente discutidos veio prover à gestão intermédia medidas que a pudessem auxiliar nas reais tarefas de gestão, como veio também traduzir em números o valor e o trabalho do IT, tarefa que antes não era tão fácil de assegurar e se tornou clara para toda a organização, fomentando um aumento de confiança e respeito pelo setor do IT.

Finalmente, um dos pontos fundamentais e decisivos para o sucesso desta iniciativa tem que ver, sem dúvida, com o apoio imprescindível da gestão de topo, como a Comissão Executiva da ESI/NBSI, do seu CEO e do próprio CIO do banco. O acompanhamento garantido pelos líderes ao longo de todo o processo, a par do empenho da liderança intermédia, provocou um comprometimento profundo nos ativos com a mudança a longo termo, do que resultou no aumento da eficiência operacional constatada.

4 Conclusão

Ainda que o projeto correspondente ao Caso A tenha sido gerido de forma tradicional, se lhe tivessem sido aplicados alguns princípios e ferramentas Lean IT ter-se-ia contribuído para otimizar a respetiva execução, sendo que determinados procedimentos da abordagem Lean são perfeitamente compatíveis com o modelo em cascata adaptado.

Na fase inicial do projeto, e para suprir a dificuldade na compreensão dos requisitos, a equipa poderia ter interagido mais frequentemente, e de forma presencial com o cliente, tirando partido, numa escala mais reduzida das vantagens de ter o cliente (ou gestor de produto) como participante ativo no projeto e da sua localização com a equipa de desenvolvimento, tal como o defende o Agile e Lean. Esta interação poderia revelar-se vantajosa para ambas as partes pois a empresa fornecedora poderia ajudar o cliente e encontrar mais facilmente aquilo que desejava no protótipo, procurando reduzir a complexidade dos seus processos e contribuindo, assim, para que se pudesse terminar o projeto dentro do tempo esperado. Além disso, se a equipa executasse reuniões diárias à semelhança do proposto pelo Lean IT, teria sido reportado logo no início à gestão a dificuldade enfrentada relativamente à falta de informação essencial para prosseguir com as tarefas respetivas. A identificação diária e constante desses constrangimentos ter-se-ia traduzido provavelmente na comunicação mais precoce de um alerta à gestão média e de topo, o que poderia implicar uma redução do atraso de execução. A falta de entendimento sobre o âmbito do projeto, as demoras nos esclarecimentos sobre os requisitos de negócio tornaram necessário alterar o código e reformular o modelo de dados, traduzindo-se em 3.288 horas de desperdício, as quais corresponderam a 18,6% do total do tempo de duração do projeto.

Ainda no que toca ao *item* da engenharia de requisitos, a equipa gastou 512 horas (2,9% do total do projeto) a desenhar e a especificar funcionalidades que acabaram por ser consideradas não prioritárias pelo cliente. Ora, segundo o Lean, os processos extras, que não trazem valor nem para o cliente nem para a equipa são consideradas desperdício e teriam sido reduzidas se, em sede de planeamento a curto e médio prazo, apenas se priorizasse as tarefas realmente importantes para as partes interessadas.

As atividades de análise poderiam ter sido facilitadas, na medida em que complexidade da sua execução seria diminuída se não houvesse necessidade de lidar

com o tratamento de todos os requisitos/funcionalidades de uma só vez, resultando desse trabalho um documento escrito de 110 páginas que deveria ser analisado e aprovado pelo cliente numa única vez. Em vez disso poder-se-ia ter dividido as funcionalidades em pequenas iterações de análise e elicitação de requisitos gerando entregáveis mais reduzidos de 10 a 20 páginas. Por um lado, uma tal divisão viabilizaria a obtenção do feedback do cliente mais rapidamente, permitindo encurtar os 132 dias úteis entre o pedido/ escrita dos requisitos e a 1ª validação do cliente na fase de testes de aceitação para vários ciclos de um mês, como defende o Lean. De facto, as entregas rápidas, por permitirem evitar acumulação simultânea de potenciais defeitos ao longo do vasto documento de análise e da quantidade de trabalho em progresso ou inacabado, que, por si constitui desperdício pois poderá tornar-se desnecessário com o passar do tempo.

Podem ainda identificar-se algumas lacunas em relação ao planeamento, tendo uma delas a ver com o facto de ter sido elaborado antes de se conhecerem os elementos que iriam constituir a equipa e o outro com pressuposto de que o cliente tinha total conhecimento da solução que pretendia. Seguindo as metodologias mais ágeis, o correto seria concretizar uma primeira iteração desenvolvendo a funcionalidade mais importante para se conhecer a cadência ótima da equipa e, com base nisso, proceder a uma projeção, mas sem nunca planear detalhadamente as atividades para um longo período de tempo como foi feito neste caso tradicional. Adicionalmente, a existência de planeamento a curto prazo, como defende o Lean e à semelhança do aplicado no caso B, torna possível ajustar o plano de forma lidar com imprevistos, recorrendo a um *buffer*, tal como a inexperiência da equipa ou a descoberta de novos requisitos antes encobertos ou desconhecidos.

Outro entrave verificado ao longo do projeto consistiu na excessiva preocupação em finalizar as atividades de acordo com o plano delineado no início do projeto, principalmente no período de análise funcional que é fechado sempre com o *sign-off*¹⁸ do documento de requisitos por parte do cliente. A priorização do seguimento do plano acima de outros interesses levou a que se forçasse o início das atividades desenvolvimento, mesmo com a análise incompleta, o que se revelou prejudicial para os analistas, cujo trabalho se viu interrompido, e para os programadores confrontados com a falta de completa informação sobre as várias funcionalidades a desenvolver. Os

¹⁸ Aprovação

programadores deram nota, aliás, da sua desconfiança no seguimento dos trabalhos, os quais, tiveram de ser refeitos mais à frente em horas extra, para não atrasar a data de entrega ao cliente, que penalizaram o tempo de duração do projeto em 1,6%.

A incerteza instalada acabou por derivar num certo grau de confusão com conseqüências na duração destas duas atividades (análise e desenvolvimento) que passaram a decorrer paralelamente, desde a primeira entrega do documento de análise. Para colmatar estes períodos de maior demanda e tentar reduzir o atraso nas atividades, revelar-se-ia necessário incorporar na equipa novos elementos, como já referido, contingência que se traduziria, num primeiro momento, em 108 horas de desperdício (0,6% do total). O incumprimento do plano revelou-se inevitável mesmo tentando seguir as datas das milestones com rigor, e terá sido por esta razão que não se tornou possível acomodar a maioria dos pedidos de alteração dos requisitos efetuados pelo cliente, pois o tempo teria sido maioritariamente gasto nas fases iniciais do projeto.

Neste projeto poderia ter-se aproveitado melhor o sistema puxado, ou *pull system*, típico do Lean para evitar pontos de restrição registados na parte de desenvolvimento referente ao *front-end*. Se o desenvolvimento tivesse sido efetuado por pequenas iterações em que a componente de *back-end* apenas entregava o respetivo produto de trabalho quando os recursos de *front-end* estivessem prontos para o receber em vez de acumula-lo em grandes pilhas sujeitando-se à acumulação de defeitos. O sistema puxado, conjugado com as reuniões diárias de acompanhamento próximo e frequente dos KPIs do projeto, teria ajudado a diminuir o desvio que resultou no desperdício de tempo dos *testers* a aguardar por novas funcionalidades a testar (correspondente a cerca de 0,5% do projeto). Também aqui, as medidas preventivas poderiam ter sido aplicadas caso se tomasse conhecimento claro da situação com a antecedência suficiente.

Uma componente do projeto geradora de grande desperdício tem a ver com a passagem a produção tardia da aplicação. Tendo por referência a perspetiva Lean, percebe-se que o fluxo contínuo de trabalho foi interrompido no momento em que o sistema completamente aprovado não passou para produção por razões externas previamente identificadas. Durante vários meses (116,4 dias úteis, correspondentes a 5,3% de tempo de projeto desperdiçado) o aplicativo terminado não seria utilizado o que resultaria em grande prejuízo visto estar-se, ainda segundo a abordagem Lean, perante trabalho inacabado que com o passar do tempo tinha largas hipóteses de se tornar obsoleto. Aqui, o Lean IT ajuda essencialmente a aumentar para quase 100% a probabilidade de entrega do trabalho na data do compromisso, como se verifica no caso

B, garantindo o cumprimento da maioria dos planos, graças ao recurso às ferramentas anteriormente analisadas e atuando na prevenção dos casos em vez da atenuação das suas consequências.

Em síntese, o projeto do caso A revelou um desperdício acumulado correspondente a cerca de 29,4% do seu tempo de duração. Confrontando-o com o caso B, conclui-se que se tivesse adotado de forma bem-sucedida procedimentos idênticos, isto é ferramentas e a metodologias Lean de forma adequada, teria poupado em desperdício ou ganho em produtividade pelo menos 23% do tempo despendidos ao longo de toda a sua duração, o que se traduz num ganho de aproximadamente 1197 horas de execução equivalente a 150 dias úteis de projeto.

Contrapondo o caso A, de uma organização não Lean, a qual revelou, no decurso do projeto, incapacidade para acompanhar o ritmo da mudança dos requisitos (ou do seu surgimento tardio e intempestivo), daí resultando uma dessincronização do IT com o negócio, ao caso B, exemplo de uma instituição que adotou a nova abordagem para contrariar os problemas que assolavam o dia-a-dia das suas equipas de IT, verificou-se que o abandono dos procedimentos tradicionais traduziu-se em ganhos de eficiência significativos.

Este caso de estudo permitiu conhecer mais a fundo uma forma de implementação Lean na componente dos sistemas de informação de um banco português, nomeadamente nas suas equipas de desenvolvimento, manutenção e de infraestruturas. O Lean concedeu à organização, como verificado, a habilidade de se adaptar ao negócio de forma rápida sem comprometer ou por em risco o IT e a qualidade dos sistemas visto que com o novo método de planeamento as equipas conseguiram acomodar alterações não planeadas de forma flexível sem prejuízo para desenvolvimentos futuros.

Através do exemplo em estudo, concluiu-se que Lean constitui uma boa abordagem quando se pretende resolver problemas associados à baixa produtividade. As equipas da ESI/NBSI criaram uma cultura de deteção e tratamento de pequenos problemas capaz de prevenir problemas maiores. Os elementos foram desde cedo incentivados a simplificar e automatizar as suas tarefas mais rotineiras, a eliminar o desperdício entendendo o fluxo de valor para o cliente, em suma, aprenderam a fazer mais com menos. Desses esforços resultou um aumento significativo da percentagem das tarefas executadas que passaram de 50-60% para 89-95%, um crescimento de eficiência de execução de tarefas para próximo dos 100% (situando-se aproximadamente nos 98%). Constatou-se,

igualmente, um aumento da produtividade que, em apenas dois anos e meio, cresceu cerca de 23%.

Com o incremento da produtividade tornou-se possível reduzir os custos e diminuir o *time-to-market*, o que se traduziu num aumento da vantagem competitiva para a empresa mãe.

A iniciativa revelou-se vantajosa nos dois aspetos Lean: o aspeto virado para o exterior, que consistiu em melhorar a satisfação do cliente, e o interior, com a melhoria do desempenho dos próprios processos do IT

5 Bibliografia

Abdullah, S., Holcombe, M. and Gheorge, M. (2006). *The impact of Agile Methodology on the wellbeing of development Teams*. Springer Science. Department of Computer Science, University of Sheffield, U.K.

Addy, R. (2007). *ITIL Weighing Pros and Cons*. Enterprise Systems Jornal

Azanha, A., Argoud, A. Junior, J., Antonioli, P (2017). *Agile Project Management with Scrum: Case Study of a Brazilian Pharmaceutical Company IT Project*. International Journal of Managing Projects in Business

Bainey, R. (2004). *Integrated IT project management: a model-centric approach*. Artech House, INC. Norwood, MA.

Balaji, S. & Murugaiyan, M. (2012) *Waterfall Vs V-Model Vs Agile: A Comparative Study On Sdlc*. International Journal of Information Technology and Business Management. Vol.2 No. 1

Beck, K., Beedle, M., Bennekum, A, Cockburn, A., Cunningham, W., Fowler, M.(2001). *The Agile Manifesto*. <http://agilemanifesto.org/>

Bell, S. (2006) *Lean Enterprise Systems: Using IT for Continuous Improvement*. John Wiley & Sons, Inc., New Jersey

Bell, S. Orzen, M. (2011) *Lean IT Enabling and Sustaining Your Lean Transformation*. Productivity Press, New York

Bennatan, E. (1995) *On Time, Within Budget Software Project Management Practices and Techniques*. John Wiley & Sons Inc. Canada

Boehm, B. (2002) *Get Ready for Agile Methods, with care*. Software Development. University of Southern California.

Boehm, B. (2006) *A View of 20th and 21st Century Software Engineering*. University of Southern California

Boehm, B. and Turner, R. (2005) *Management Challenges to Implementing Agile Processes in Traditional Development Organizations* IEEE Software 2005. IEEE Computer Society

Boehm, Barry and Turner, Richard, *Balancing Agility and Discipline—A Guide for the Perplexed*. New York: Addison-Wesley, 2003, pp. 36–37

Castro, R. (2012). *Lean Six Sigma, Para Qualquer Negócio*. 1ª Edição, IST Press. Lisboa

Cobb, C (2011). *Making Sense of Agile Project Management: Balancing Control and Agility*. John Wiley & Sons, Inc.

Connors, D.(1992) *Software Development Methodologies and Traditional and Modern Information Systems*. ACM SIGSOFT SOFTWARE ENGINEERING NOTES vol 17 no 2

Daniel J. Fernandez & John D. Fernandez (2008) *Agile Project Management—Agilism versus Traditional Approaches*, Journal of Computer Information Systems, 49:2, 10-17

Davis, P. e Bentley, W. (2010) *Six Sigma Secrets for de CIO*. CRC Press New York

Fibra (2011). *Aumentar a produtividade com o Lean IT*. Revista Especializada Mensal. Portugal

Hass, K. (2007), *The Blending of Tradicional and Agile Project Management*, PM World Today (Vol- IX, Issue V)

Hibbs, C., Jewett, S. & Sullivan, M. (2009). *The Art of Lean Software Development*. O'Reilly Media, Inc., California.

Highsmith, J (2002) *Agile Software Development Ecosystems*. Addison Wesley

Hoda, R., Noble, J. and Marshall, S. (2011) *Developing a grounded theory to explain the practices of self-organizing teams*. Springer Science. New Zealand

Janes, A., Succi, G. (2014) *Lean Software Development in Action*. Springer-Verlag Berlin Heidelberg 2014

Ji, F. & Sedano, T. (2011) *Comparing Extreme Programming and Waterfall Project Results*. IEEE, Carnegie Mellon University, Silicon Valley Campus.

Jugulum, R., Samuel, P. (2008). *Design for Lean Six Sigma A Holistic Approach to Design and Innovation* John Wiley & Sons, Inc. New Jersey

Kasse, T. (2004) *Practical Insight into CMMI®*. Artech House Boston • London

Kumar, D. (2005). *Lean Software Development*. The project perfect – White Paper Collection.

Layton, M. (2012) *Agile Project Management For Dummies*. Paperback

Luckey, T., Phillips, J. (2006). *Software Project Management for Dummies*. Wiley Publishing, Inc. Indiana.

Martin, J. (2010) *Measuring and Improving Performance Information Technology Applications in Lean System*. Productivity Press, NY

Maruping, L., Venjatesh, V. and Agarwal, R. (2009) *A Control Theory Perspective on Agile Methodology Use and Changing User Requirements*. Information Systems Research. University of Arkansas

Miguel, A. (2015) *Gestão de Projetos de Software*. (5ªed., pp 35-36). Lisboa: FCA Editora de Informática.

Munassar, N. & Govardhan, A (2010). *A Comparison between Five Models of Software Engineering*. International Journal of Computer Science Issues, Vol. 7, Issue 5.

Petersen, K., Wohlin, C. & Baca, D. (2009) *The Waterfall Model in Large-Scale Development*. Springer-Verlag Berlin Heidelberg, pp. 286-400

PMI (2012), *A Guide to the Project Management Body of Knowledge (PMBOK® Guide) Fifth Edition*, Project Management Institute, Newton Square, PA

Poppendieck, M. Poppendieck, T. (2013). *The Lean Mindset Ask the Right Questions*. Addison-Wesley, Westford, Massachusetts.

Poppendieck, M., Cusumano, M. (2012). *Lean Software Development: A Tutorial*. IEEE Comput. Soc.

Poppendieck, M., Poppendieck, T., (2003). *Lean Software Development: An Agile Toolkit*. Addison Wesley, Boston.

Rakos, J. (1990) *Software Project Management For Small to Medium Sized Projects*. PRENTICE HALL, Englewood Cliffs, New Jersey.

Schmidt, J. & Lyle, D. (2010). *Lean Integration: An integration Factory Approach to Business Agility*. Addison-Wesley. New York

SEPG (2008). *Best of Everything – ITIL, CMMI & Lean Six Sigma*. Tampa, FL

Shelton, C. (2008). *Agile and CMMI: Better Together*. Scrum Alliance.

Singh, M. (2008) *U-SCRUM: An Agile Methodology for Promoting Usability*. AGILE'08 Conference, Toronto.

Sommerville (2011) *SOFTWARE ENGINEERING*. (Ninth Edition). Boston: Pearson Education, Inc., publishing as Addison-Wesley.

- Staats, B., Brunner, D., Upton, D. (2010) *Lean principles, learning, and knowledge work: Evidence from a software services provider*. Elsevier, CA.
- Stellman, A., Greene, J. (2015) *Learning Agile. Understanding Scrum, XP, Lean and Kanban*. O'Reilly Media, Inc., California
- Stellman, A., Greene, J. (2005). *Applied Software Project Management*. O'Reilly. Sebastopol, California.
- Sutherland (2016). *A Arte de fazer o dobro do trabalho em metade do tempo*. (1ª ed., p.13). Lisboa: Lua de papel (Originalmente publicado em 2014)
- Sutherland, J. & Ahmad, N. (2011). *How a Traditional Project Manager Transforms to Scrum: PMBOK vs. Scrum*. Presented paper at Agile 2011
- Sutherland, J. (2013). *Agile Principles and Values White Paper* [https://msdn.microsoft.com/en-us/library/dd997578\(v=vs.190\).aspx](https://msdn.microsoft.com/en-us/library/dd997578(v=vs.190).aspx)
- Sutherland, J. Schwaber, K. (2007) *The Scrum Papers: Nuts, Bolts, and Origins of an Agile Process*
- The Standish Group (2016). *Chaos Report*. The Standish Group International, Inc
- Uikey, N., Suman, U. & Ramani, A., (2011). *A Documented Approach in Agile Software Development*. International Journal of Software Engineering (IJSE), Volume (2), Issue (2)
- Waterhouse, P. (2008). *Lean IT: "Waste Not, Want Not" Strategies to Reduce Eight Elements of Waste in IT*. CA Transforming IT Management
- Womack, J., Jones, D., Roos, D. (2007). *The Machine That Changed The World – How Lean Production Revolutionized the Global Car Wars*. UK: Simon & Schuster
- Yin, R. K. (2002). *Case Study Research, Design and Methods*, 3rd ed. Newbury Park, Sage Publications.
- Zak, A. & Waddell, B. (2011). *Organizing and Aligning the Management Team in a Lean Transformation*. Productivity Press, New York.
- Ziółkowski, A. & Deręgowski, T. (2014) *Hybrid Approach in Project Management – Mixing Capability Maturity Model Integration with Agile Practices*. SSN 1392–0758 SOCIAL SCIENCES. Nr. 3 (85). Poland.