



INSTITUTO SUPERIOR TÉCNICO
Universidade Técnica de Lisboa

UMA FERRAMENTA COLABORATIVA PARA A GESTÃO DE PEQUENOS PROJECTOS DE SISTEMAS DE INFORMAÇÃO

Bruno Miguel dos Santos Costa Domingos

Dissertação para Obtenção de Grau de Mestre em

Engenharia Informática e de Computadores

Júri

Presidente: Professor Alberto Manuel Rodrigues da Silva

Orientador: Professor Miguel Leitão Bignolas Mira da Silva

Vogal: Professora Maria do Rosário Gomes Osório Bernardo Ponces de
Carvalho

Setembro 2007

Agradecimentos

Gostava de agradecer ao Professor Doutor Miguel Mira da Silva pelas oportunidades cedidas e pelo seu empenho exemplar como orientador.

Aos meus pais António e Isabel, pois sem eles não teria sido possível chegar até aqui e aos meus irmãos, Rita e Pedro, pela paciência e pelo carinho.

À Mafalda pela amizade, pelo apoio, pela ternura, pela força, pela compreensão. Não teria conseguido sem ti...

A todos os meus amigos e colegas que me acompanharam neste período da minha vida, um forte abraço.

Resumo

Gerir e liderar estruturas organizacionais orientadas a projectos é diferente do modelo comum da maioria das organizações. Porém, hoje em dia, este paradigma é cada vez mais utilizado em todo o mundo, principalmente nas empresas prestadoras de serviços de SI / IT.

O problema abordado nesta tese consiste na reduzida taxa de sucesso que os projectos de SI / TI têm alcançado ao longo dos anos, especialmente os pequenos projectos. Estes apresentam características específicas de tempo e custos que os diferenciam dos projectos de grande escala e, conseqüentemente, requisitos próprios no planeamento e na comunicação que se reflectem, inevitavelmente, no sucesso das metodologias aplicadas e nas ferramentas de software (SW) utilizadas.

Com o objectivo de contribuir para a resolução de alguns dos problemas inerentes a esta temática, a proposta desta tese encontra-se dividida nestes dois pontos de interesse, metodologias e ferramentas, apresentando-se como uma alternativa para gerir pequenos projectos de sistemas de informação.

A proposta materializa-se de acordo com um modelo desenhado à imagem das necessidades dos pequenos projectos e uma ferramenta de SW. Esta ferramenta implementa uma lista de requisitos levantados através da análise dos pontos de falha das ferramentas de SW de gestão de projectos mais utilizadas e foi testada em ambiente empresarial com resultados bastante satisfatórios.

Palavras-Chave

Gestão de Projectos, Pequenos Projectos de Sistemas de Informação, Comunicação, Planeamento e Controlo, Ferramentas de Suporte, Metodologias, Ambiente Colaborativo.

Abstract

Managing and leading project oriented organizational structures is different from the common model of most organizations. However, nowadays, this paradigm has become more practiced worldwide, mainly in IT services enterprises.

In this thesis, the main problem deals with the tiny success rate that IT projects showed in the last few years, specially the small projects. This kind of project presents specific features of time and cost that differentiates them from the large projects. Consequently, dissimilar requirements of planning and communication impact, inevitably, the success of the methodologies applied and the software (SW) tools.

This thesis's goal is to help with the resolution of some of these problems. The proposal is based on two key points: methodologies and SW tools. It presents itself as an alternative way of managing small information systems projects.

This proposal was materialized through a model designed according to the needs of the small projects and a SW tool. This tool was implemented to fulfill the requirements gathered by the analysis of the faults of most used project management SW tools and it was tested in a competitive environment with satisfactory results.

Keywords

Project Management, Small Projects of Information Systems, Communication, Planning and Controlling, Support Tools, Methodologies, Collaborative Environment.

Índice

Agradecimentos.....	III
Resumo.....	V
Palavras-Chave	V
Abstract.....	VI
Keywords	VI
Índice	VII
Índice de Figuras	X
Índice de Tabelas	XIII
Definições e Acrónimos.....	XIV
1. Introdução.....	1
1.1. Contexto	1
1.2. Problema	2
1.3. Proposta	2
1.4. Organização do documento	3
2. Gestão de Projectos	5
2.1. Projecto	5
2.2. Gestão de Projectos.....	5
2.3. Metodologias e Ferramentas.....	8
2.3.1. Metodologias	9
2.3.1.1. Capability Maturity Model.....	9
2.3.1.2. Virtual Members / Teams.....	11
2.3.1.3. Inspection Method.....	12
2.3.1.4. eXtreme Programming.....	13
2.3.1.5. Rational Unified Process (RUP).....	15
2.3.1.6. PRINCE 2.....	16
2.3.1.7. Processos de desenvolvimento projectos de software.....	19
2.3.1.8. Scrum.....	21
2.3.1.9. Feature Driven Development (FDD)	22
2.3.1.10. Dynamic System Development Method (DSDM).....	24

2.3.1.11.	Adaptative Software Development (ASD).....	24
2.3.1.12.	Conclusão	25
2.3.2.	Ferramentas	26
2.3.2.1.	Microsoft Project	27
2.3.2.2.	Quickbase	29
2.3.2.3.	Basecamp	29
2.3.2.4.	Conclusão	30
3.	Pequenos Projectos.....	33
3.1.	Projectos de Sistemas de Informação	33
3.2.	Gestão de Pequenos Projectos	35
3.2.1.	Gestor de Projecto.....	35
3.2.2.	Características.....	36
3.2.3.	Fase de Desenho	38
3.3.	Metodologias e Ferramentas.....	39
3.3.1.	Metodologias	39
3.3.2.	Ferramentas	40
3.4.	Problemas da Gestão de Pequenos Projectos	41
3.5.	Conclusão.....	45
4.	Proposta.....	47
4.1.	Metodologia.....	47
4.2.	Requisitos da Ferramenta	51
4.2.1.	Perfis de Utilizador.....	51
4.2.2.	Modelo de Domínio.....	52
4.2.3.	Requisitos Funcionais.....	52
4.3.	Processos.....	54
4.3.1.	Criar um Pequeno Projecto	54
4.3.2.	Fecho de um Pequeno Projecto	56
4.3.3.	Criar Nova Tarefa	57
4.3.4.	Terminar Tarefa	59
5.	Implementação	61
5.1.	Processo de Desenvolvimento.....	61
5.2.	Protótipo	62
6.	Avaliação	75
6.1.	Tecnologia.....	76
6.2.	1º Protótipo.....	77

6.3.	2º Protótipo.....	79
6.4.	Conclusão.....	79
7.	Conclusão.....	81
7.1.	Trabalho futuro.....	83
	Referências.....	85

Índice de Figuras

Figura 1. Relação entre Qualidade, Custo e Tempo (Fonte: [5]).....	6
Figura 2. Fases do ciclo de vida de um projecto (Fonte: [27]).....	8
Figura 3. Estrutura e relações da CMM (Fonte: [28])	10
Figura 4. Fluxo típico de iterações da RUP (Fonte: [33]).....	15
Figura 5. Diagrama geral da RUP (Fonte: [8])	16
Figura 6. Processos que constituem a metodologia PRINCE2 (baseado em [35] e [36])....	18
Figura 7. Processo da <i>Scrum</i> [11].....	22
Figura 8. Fases do Processo do <i>Feature Driven Development</i> [46].....	23
Figura 9. Análise das Metodologias (Área do círculo – período de implementação; Cor do círculo – recursos necessários).....	26
Figura 10. Microsoft Project	27
Figura 11. Quickbase	29
Figura 12. Basecamp	30
Figura 13. Análise das Ferramentas (Área do círculo – Funcionalidades da planeamento e controlo; Cor do círculo – Funcionalidades de gestão de recursos).....	31
Figura 14. a) Actividades de gestão que contribuem para o insucesso dos projectos de TI (Fonte: [62]); b) Fases de insucesso (Fonte: [62]); c) Causas de insucesso (Fonte: [62]); d) Factores críticos de sucesso (Fonte: [62])	33
Figura 15. Metodologia proposta	50
Figura 16. Criação de um Pequeno Projecto (proposta efectuada por um utilizador da aplicação).....	54
Figura 17. Criação de um Pequeno Projecto (proposta efectuada pelo administrador da aplicação).....	55
Figura 18. Fecho de um Pequeno Projecto	56
Figura 19. Criação de Nova Tarefa (Gestor de Projecto)	57

Figura 20. Criação de Nova Tarefa (Proposta de Membro da Equipa de Projecto).....	58
Figura 21. Terminar Tarefa	59
Figura 22. Processo de Desenvolvimento de Software utilizado.....	61
Figura 23. Ecrã de Autenticação na Aplicação	62
Figura 24. <i>Dashboard</i> – Informação geral do utilizador (RF10)	63
Figura 25. Proposta de novo projecto.....	63
Figura 26. <i>Gantt Chart</i> de Projectos (RF3, RF4, RF10, RF11, RF13).....	64
Figura 27. Ecrã de Gestão de Mensagens (RF8)	65
Figura 28. Calendário de tarefas individual para cada utilizador (RF1, RF10, RF11, RF12, RF13).....	65
Figura 29. Vista geral de um projecto (RF1, RF3, RF5, RF7, RF9, RF10, RF11).....	66
Figura 30. Grelha de tarefas de um projecto (RF1, RF4, RF10, RF11, RF13).....	67
Figura 31. Mecanismo de gestão de riscos (RF6)	68
Figura 32. Mecanismo de gestão de <i>issues</i> (RF6)	68
Figura 33. Fóruns do projecto (RF8).....	69
Figura 34. Equipa de Projecto.....	69
Figura 35. Ecrã geral de uma tarefa (RF1, RF3, RF5, RF7, RF11).....	70
Figura 36. Mecanismo de gestão de <i>deliverables</i> (RF5)	71
Figura 37. Exemplo de notificação enviada pelo sistema (RF14)	71
Figura 38. Gestão de propostas de projectos (RF2).....	72
Figura 39. Gestão de projectos que estão a ser geridos com auxílio da ferramenta (RF2). 72	
Figura 40. Gestão dos dados dos utilizadores da aplicação	73
Figura 41. Plataforma tecnológica da Arquitectura OutSystems	77
Figura 42. Vista semanal do Plan Grid	79

Índice de Tabelas

Tabela 1. Diferença entre Projectos e Operações (Fonte: [5])	5
Tabela 2. Satisfação com ferramentas de gestão de projectos [24] (escala 0-5)	28
Tabela 3. Comparação entre ferramentas	31
Tabela 4. Problemas básicos da gestão de múltiplos pequenos projectos (Fonte: [20])	42
Tabela 5. Requisitos Funcionais de uma ferramenta de software de gestão de pequenos projectos	52
Tabela 6. Avaliação da proposta por comparação com as ferramentas analisadas	75

Definições e Acrónimos

ASD – Adaptative Software Development

CMM – Capability Maturity Model

DSDM – Dynamic System Development Method

FDD – Feature Driven Development

IT – Information Technologies

KPA – Key Process Area

RUP – Rational Unified Process

SCM – Software Configuration Management

SEP – Software Engineering Process

SI – Sistemas de Informação

SQA – Software Quality Assurance

SW – Software

WBS – Work Breakdown Structure

XP – eXtreme Programming

1. Introdução

Este documento apresenta um trabalho desenvolvido na área de gestão de projectos, no sentido de perceber quais os factores que tornam a taxa de insucesso desta disciplina tão alta, especialmente na área dos pequenos projectos de sistemas de informação.

1.1. Contexto

Um projecto é uma actividade temporária composta por um conjunto bem definido de objectivos de negócio e resultados esperados. Como tal, para ter sucesso e para que se garanta que os objectivos de um projecto são cumpridos, é necessário que haja um controlo, um rumo para o projecto. Esse rumo é assegurado através de um conjunto de procedimentos normalmente estruturados e seguidos pela equipa do projecto. A esta disciplina dá-se o nome de gestão de projectos [1] [2] [3] [4].

A gestão de projectos tem como objectivo garantir que um projecto seja concluído respeitando as normas de custo, tempo e qualidade inicialmente definidas. Esta disciplina está dividida em várias áreas da gestão e é suportada por um conjunto de práticas com o objectivo de garantir o sucesso desta actividade.

As áreas que devem ser contempladas pela gestão em qualquer projecto são [5]:

- Gestão do âmbito
- Gestão da organização
- Gestão da qualidade
- Gestão de recursos
- Gestão de custos
- Gestão do tempo
- Gestão de riscos

Não existe um mapeamento directo destas áreas em todos os tipos de projecto. Sempre que se começa um novo projecto é necessário implantar todos estes mecanismos à medida das necessidades e estruturá-lo segundo a lógica do projecto em causa.

A forma mais comum de o fazer é seguindo a seguinte abordagem [5] [4]:

- Arranque
- Planeamento
- Controlo
- Execução
- Avaliação

- Encerramento

O passo seguinte é dividir o projecto em actividades e tarefas segundo cada uma destas fases e, atribuir responsabilidades e reservar os recursos necessários a cada uma das actividades planeadas.

Para além dos procedimentos referidos, existem um conjunto de metodologias e ferramentas com o objectivo de potenciar a correcta prática da gestão de projectos.

1.2. Problema

As metodologias existentes, como por exemplo, o *Capability Maturity Model* (CMM) [6] [7], *Rational Unified Process* (RUP) [8] [9] [10], *eXtreme Programming* (XP) [11] [12] [13], PRINCE2, *Virtual Members* [14] [15] [16] ou *Inspection Method* [17] [18] [19] produzem resultados satisfatórios quando aplicadas a grandes projectos de software. Porém, em casos mais específicos como o de pequenos projectos de SI (sistemas de informação) / TI (tecnologias de informação), os resultados mostram que não é bem assim [20].

Analogamente, as ferramentas de software, que são hoje em dia indispensáveis para suportar a gestão de projectos, partilham o mesmo problema [20]. A maioria delas é satisfatória na gestão de grandes projectos [21] [22] [23] [24], onde os gestores estão já especializados e treinados nessas ferramentas, cuja complexidade é elevada.

Mas, os pequenos projectos têm características particulares a vários níveis que impossibilitam que as ferramentas e metodologias existentes possam surtir bons resultados.

As características dos pequenos projectos mais relevantes que marcam esta diferença são [20] [25] [26]:

- Tempos de execução curtos
- Comunicação forte
- Recursos partilhados
- Ambiente multi-projecto
- Fase de desenho característica

1.3. Proposta

O trabalho resultante desta investigação foi, no seguimento dos problemas mencionados, levantar os requisitos que os pequenos projectos de SI / TI apresentam, perceber os pontos de falha que as metodologias e ferramentas existentes apresentam, e propor uma ferramenta capaz de responder às necessidades levantadas, alinhada com uma metodologia à medida.

A proposta materializou-se numa ferramenta de software desenhada segundo o conjunto de requisitos levantado e foi testada em ambiente empresarial por uma equipa da área de sistemas de informação.

1.4. Organização do documento

No capítulo seguinte, encontra-se o capítulo Gestão de Projectos que corresponde à área onde se enquadra esta tese. Nele é apresentada uma definição do que é um projecto e a disciplina de gestão de projectos, bem como as abordagens mais comuns utilizadas quando se gere um projecto. Na parte final do capítulo encontra-se a análise do estado da arte relativo às metodologias e ferramentas de software de suporte à gestão de projectos e um breve levantamento de alguns problemas que estas apresentam.

O capítulo 3 apresenta os problemas relativos ao insucesso dos projectos de sistemas de informação. De seguida, é introduzida a definição de pequenos projectos e as características que os distinguem dos projectos de grande escala. Por fim encontra-se o resumo dos problemas apresentados pelas metodologias e ferramentas existentes desenhados para grandes projectos e não aplicáveis a pequenos projectos.

No capítulo 4 encontra-se a proposta desta tese dividida entre uma metodologia e uma ferramenta à imagem das necessidades dos pequenos projectos. No capítulo seguinte, capítulo 5, é explicado como esta proposta poderá ser implementada através de um caso concreto de uma ferramenta que obedece aos requisitos descritos.

A avaliação da ferramenta é efectuada no capítulo 6 através da descrição da Tecnologia, do período de execução e de testes relativos ao 1º e 2º protótipos e a validação dos objectivos propostos e alcançados.

2. Gestão de Projectos

2.1. Projecto

Como projecto entende-se: “Um esforço complexo, destinado a atingir um objectivo específico, dentro de determinado prazo e orçamento, o qual tem normalmente uma natureza multi-funcional, é único e não repetitivo dentro da Organização”, Cleland and King (1983).

Atendendo à definição apresentada no parágrafo anterior, é necessário e muito importante distinguir um projecto de uma operação para que estas não sejam interpretadas e/ou geridas como projectos. As operações são a base da pirâmide nas empresas e, como tal, estão já bem definidas e implantadas para que as organizações possam funcionar em pleno. Por outro lado os projectos são instáveis e bastante propícios a riscos.

A Tabela 1 ilustra os parâmetros que distinguem estas duas abordagens [5]:

Tabela 1. Diferença entre Projectos e Operações (Fonte: [5])

Projectos	Operações
Singularidade	Repetitividade
Finitos	Eternos
Mudança Revolucionária	Mudança Evolucionária
Recursos Transitórios	Recursos Estáveis

2.2. Gestão de Projectos

Tendo por base várias definições de gestão de projectos existentes na literatura, é possível definir a gestão de projectos como uma disciplina que tem como principais objectivos gerir a correcta execução de um projecto garantindo o seu sucesso através da aplicação de um conjunto de ferramentas e técnicas de modo a gerir os recursos na realização de tarefas complexas e com restrições temporais, de custo e qualidade [1] [2] [3] [4]. Esta disciplina

encontra-se formalmente dividida em várias sub-disciplinas que, no seu conjunto, contribuem para o já referido objectivo da gestão de projectos. Essas sub-disciplinas são [5]:

- Gestão do Âmbito – definição do projecto, a sua finalidade e requisitos.
- Gestão da Organização – todas as partes envolvidas.
- Gestão da Qualidade – especificações.
- Gestão de Recursos – alocações de acordo com as necessidades.
- Gestão de Custos – realização do âmbito de acordo com a qualidade.
- Gestão do Tempo – período de realização.
- Gestão de Riscos – riscos associados ao projecto, bem como planos de contingência.

Cada um dos tópicos anteriores deve ser tido em conta sempre que se prepara o arranque de um projecto, ainda que, dependendo do objectivo final do mesmo ou da cultura da organização, alguns deles possam já estar bem definidos *a priori*, tendo por isso que ser seguidos pelas equipas de projecto.

Existe um modelo que relaciona algumas destas variáveis e que demonstra a directa relação entre elas.

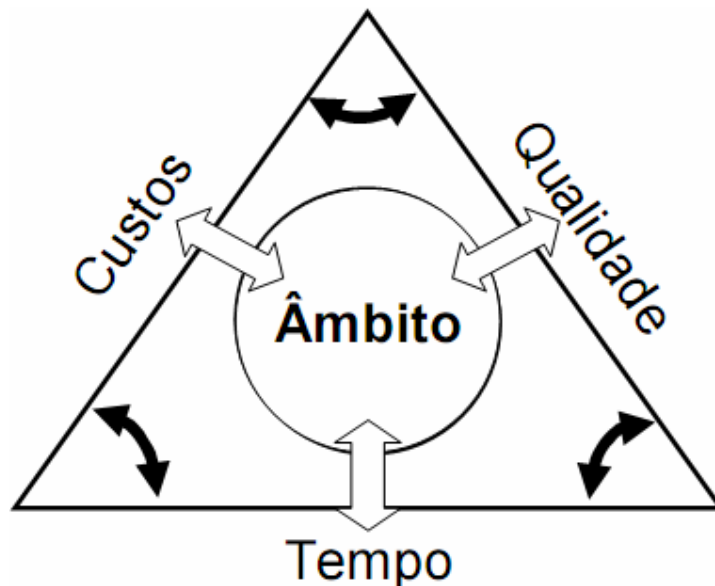


Figura 1. Relação entre Qualidade, Custo e Tempo (Fonte: [5])

Da Figura 1 retiram-se algumas considerações que devem acompanhar qualquer equipa de projecto no desenvolvimento do mesmo. O âmbito de um projecto está condicionado pela sua Qualidade, pelo seu Custo e pelo Tempo em que tem que ser executado. Como tal, deve evitar-se desequilibrar o triângulo, mantendo a coerência das três dimensões, de modo a que, se o projecto tem que ser feito num dado período de Tempo, a sua Qualidade deve manter-se o mais realista possível de modo a não aumentar a dimensão Custo de modo incontrolado. Se,

pelo contrário a dimensão chave for a Qualidade, então deve ponderar-se bem o Tempo de execução do Projecto com o mesmo fim de controlar os Custos.

Para além desta divisão, um projecto pode ainda ser visto como um conjunto de fases generalista e, portanto, comum a qualquer instância. As fases mais comuns que compõem um projecto são [5] [4]:

- Arranque e Planeamento
 - Formulação do projecto
 - Identificação dos objectivos de negócio e constrangimentos
 - Delimitação do âmbito e organização do projecto
 - Identificação dos riscos e pressupostos
 - Elaboração da *Work Breakdown Structure* (WBS)
 - Contratação (se necessário)
- Calendarização e Controlo
 - Tempo, *milestones*, *deadlines* e *deliverables*
 - Orçamento
 - Recursos
- Execução
- Avaliação e Encerramento
 - Medir custos e tempos
 - Impactos de alterações
 - Fecho do projecto

O Arranque do projecto surge, cronologicamente, em primeiro lugar. Esta fase é preponderante na continuidade do projecto, pois nela é feita uma análise de viabilidade do mesmo. No entanto, esta tarefa pode ser ignorada se as ordens da gestão de topo forem no sentido do projecto se realizar obrigatoriamente.

A fase de Calendarização e Controlo é transversal a todo o projecto. É muito comum fazerem-se revisões às calendarizações devido a atrasos de algumas tarefas que podem ter impactos no caminho crítico de um projecto.

O Controlo é, por natureza, uma actividade recorrente. É usual que o gestor de projecto reserve um período semanal, dependendo da dimensão do projecto, para se dedicar a esta tarefa que envolve actividades como actualização das tarefas concluídas e reavaliação do esforço.

A Execução do projecto corresponde ao período onde este é realmente implementado. É durante esse intervalo que o gestor de projecto deve efectuar o planeamento e o controlo do mesmo. É também nesta fase que ocorrem os comuns atrasos nos projectos e onde os

gestores são obrigados a alocar mais recursos para os prazos serem cumpridos. Este cenário é facilmente visível no gráfico da Figura 2:



Figura 2. Fases do ciclo de vida de um projecto (Fonte: [27])

Avaliação e Encerramento é uma fase bastante importante na gestão de projectos. É nela que se retiram as conclusões mais importantes de acordo com o que correu bem e mal num projecto. Tipicamente, toda a equipa de projecto deverá participar nesta tarefa de modo a que todos os aspectos sejam levantados e percebidos por todos de modo a que, em projectos futuros, se possa tirar partido das várias experiências passadas.

A par destes modelos existe um conjunto de outras premissas que devem ser cumpridas nesta árdua função da gestão de projectos [5].

- Qualquer projecto deve ser gerido segundo uma base sólida construída no início do mesmo, uma estrutura de trabalho: WBS.
- O foco da gestão deve centrar-se em modelos “o que obter” e em “resultados” e não em “como fazer”.
- Definir no início as responsabilidades de todas as partes envolvidas, os seus papéis e relações.
- Avaliar os recursos disponíveis, tecnologias a utilizar e impactos/mudança na organização.

2.3. Metodologias e Ferramentas

De modo a auxiliar as equipas de projecto a desempenhar correctamente os seus papéis, foram-se desenvolvendo ao longo dos tempos nas empresas de todo o mundo uma série de metodologias e ferramentas. Neste capítulo, são apresentadas parte destas metodologias e são analisadas algumas ferramentas na forma como se adequam a pequenos projectos.

2.3.1. Metodologias

Ao longo dos tempos, as empresas sentiram necessidade de criar processos de auxílio à gestão e implementação de projectos devido aos problemas resultantes do modo como estes são geridos. Neste sentido, foram surgindo metodologias e processos aplicáveis a pequenos projectos ou adaptáveis aos mesmos visto que, na sua origem, muitos destes processos foram pensados para projectos de larga escala. Contudo, a maioria destes processos foca-se em projectos específicos, na maioria de desenvolvimento de software, e não se assumem como genéricos e aplicáveis a qualquer tipo de pequenos projectos, nomeadamente a pequenos projectos de sistemas de informação.

De seguida, são referidas as metodologias mais usadas para tentar resolver ou minimizar alguns dos problemas associados à gestão de projectos, acompanhadas de uma breve explicação.

2.3.1.1. Capability Maturity Model

O CMM [6] [7] é um modelo baseado em boas práticas de desenvolvimento de software (SW) em larga escala que tem como principal objectivo desenvolver e refinar os processos de uma organização em áreas como engenharia de SW e de sistemas, gestão de projectos, gestão do risco, tecnologias de informação e gestão de recursos. Este está estruturado segundo cinco níveis de maturidade [28] [10]:

- *Initial*: os processos são *ad-hoc* e a organização não proporciona um ambiente estável.
- *Repeatable*: as boas práticas de projectos anteriores são repetidas.
- *Defined*: neste nível são definidos processos normalizados de modo a garantir consistência dentro da organização.
- *Managed*: utilizam-se medidas bem definidas de modo a que a gestão possa, efectivamente, controlar o esforço no desenvolvimento de SW.
- *Optimizing*: este nível foca-se em otimizar continuamente a performance dos processos através de melhorias tecnológicas inovadoras e incrementais.

À excepção do primeiro nível, estão definidos para os restantes os chamados *key process areas* (KPA). Cada KPA é composto por actividades relacionadas entre si que, quando executados colectivamente, cumprem os objectivos estipulados para cada área. Exemplos de KPA são *Project Monitoring and Control*, *Project Planning* ou *Supplier Agreement Management*.

Os objectivos de cada KPA sumarizam os estados que devem existir para garantir que cada um deles seja executado de forma eficaz e duradoura. Consoante os objectivos atingidos

e com base em indicadores, é possível determinar o nível de uma organização. Resumidamente, os objectivos delimitam o âmbito e o propósito de cada KPA.

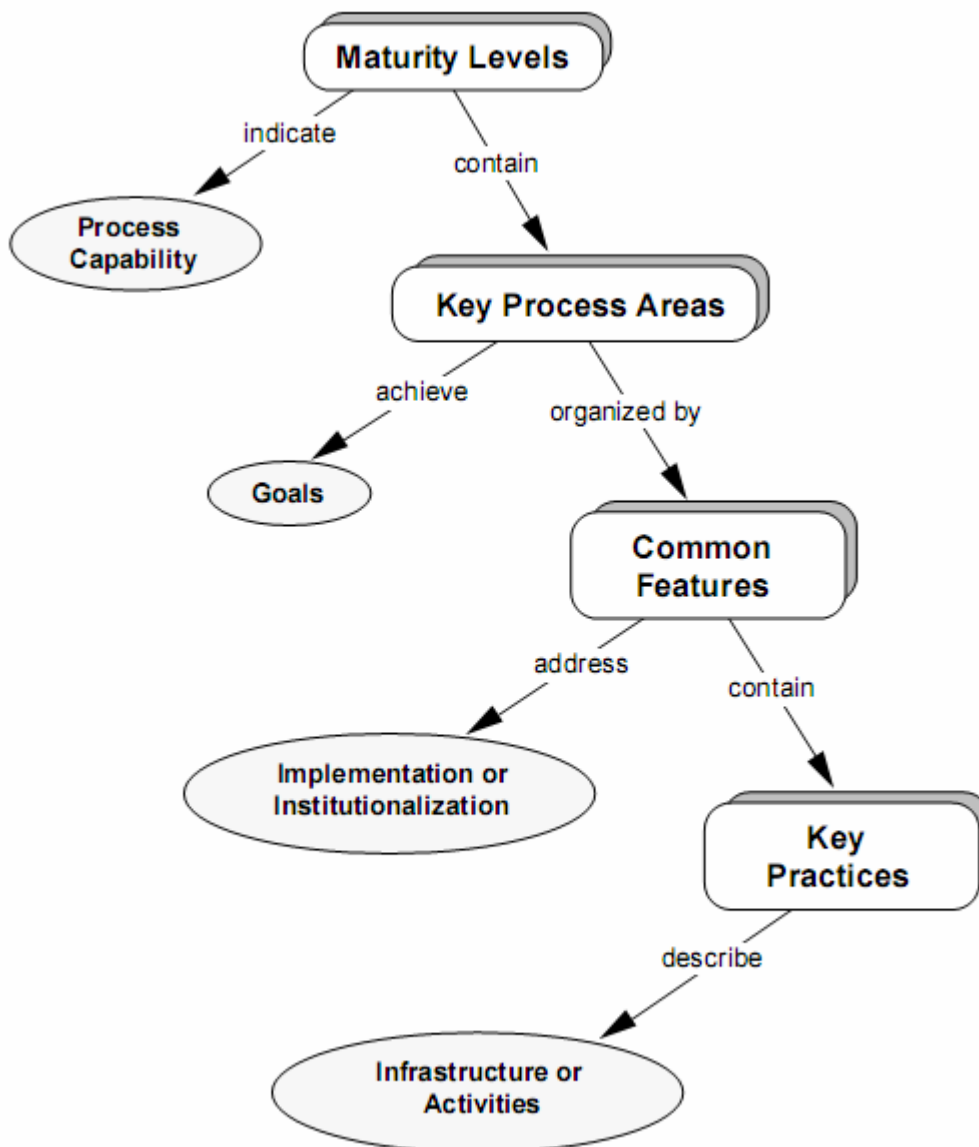


Figura 3. Estrutura e relações da CMM (Fonte: [28])

Na Figura 3 encontra-se representada a relação existente entre os níveis de maturidade e os KPA. Por fim, as práticas (*key practices*) que implementam concretamente um KPA são agrupadas segundo cinco conjuntos que dão de nome:

- *Commitment to Perform*
- *Ability to Perform*
- *Activities Performed*
- *Measurement and Analysis*

- *Verifying Implementation*

Todavia, este modelo não pode ser aplicado de forma directa a pequenos projectos de sistemas de informação pois acarreta uma série de problemas:

- Excesso de documentação
- Estrutura de gestão não relacionada
- Espaço incapaz para revisões
- Níveis elevados de requisição de recursos
- Elevados custos de treino
- Falta de orientação
- Práticas não relacionadas

No entanto, Paulk [29] propõe alterações ao processo de modo a adaptar esta metodologia aos pequenos projectos. Estas modificações passam por:

- Reduzir o número de fases de modo a simplificar o fluxo de tarefas,
- Minimizar a documentação,
- Manter a qualidade.

De qualquer modo existiram sempre custos associados a esta adaptação:

- Custos de treino,
- Custos de especialização de sub-equipas orientadas para sub-processos independentes (Software Configuration Management (SCM), Software Quality Assurance (SQA) e Software Engineering Process (SEP))
- Necessidades de adaptação das práticas KPA que não estão relacionadas com pequenos projectos.

Em suma, esta metodologia apresenta-se desapropriada para pequenos projectos, nomeadamente para projectos de sistemas de informação, pois implica treino e especialização por parte dos elementos da equipa de desenvolvimento para exercerem funções específicas e custos de adaptação dos sub-processos SCM, SQA e SEP dado que estes não foram desenhados para pequenos projectos.

2.3.1.2. Virtual Members / Teams

Este é um modelo aplicado em ambientes de múltiplos projectos onde os recursos são limitados [14] [15] [16]. O ponto essencial desta metodologia é garantir um meio de comunicação óptimo de modo a que o gestor de projecto possa gerir um número largo de projectos sem que haja necessidade de parar com a realização das suas tarefas e das dos restantes elementos.

Os elementos das várias equipas mantêm, on-line, guias e *templates* de manuais baseados nas suas experiências. Além disso, as equipas mantêm ainda toda a informação relativa ao estado do projecto, riscos, *issues* e questões on-line. Quando um elemento de outra equipa necessita dos conhecimentos de outros elementos cujo projecto primário não é o seu, requisitam esses elementos como *Virtual Members* [30].

Os *Virtual Members* dedicam uma parte do seu tempo para rever a documentação de projectos secundários fornecida pelo elemento que o requisitou e dão respostas e recomendações. Existe uma maior cooperação entre os elementos das equipas geridas pelo gestor dos vários projectos num determinado momento.

Esta metodologia é eficaz na cooperação entre equipas, no entanto os elementos necessitam de canalizar algum para a revisão do trabalho dos outros, o que pode ter impactos negativos na calendarização dos vários projectos. Para além disso, a metodologia não dá resposta a todas as áreas e, a comunicação, não é o único problema dos pequenos projectos.

2.3.1.3. Inspection Method

Um método de inspecção é um controlo de qualidade para produtos acabados [17]. É usado maioritariamente em projectos de larga escala e necessita de 3 a 8 pessoas para o fazer. Runge [18] defende que este processo faz sentido também em pequenos projectos e é aplicável aos mesmos desde que existam determinadas condições. Para justificar a aplicação deste processo a pequenos projectos, o autor baseia-se em parte no comentário de outro autor, Fagan [19], que diz o seguinte: “o custo de se descobrirem e corrigirem os erros cedo, é de 10 a 100 vezes menor do que descobrir e corrigir os erros na fase de testes do projecto”.

O processo mais comum de inspecção consiste nos passos seguintes [31]:

- *Ovierview* – fase onde são descritas as várias dimensões do projecto e onde é dada formação para o grupo de agentes que vão concretizar o processo.
- *Preparation* – fase de formação ao nível do indivíduo. Cada participante estuda o desenho, o seu papel e a sua lógica. É muito comum rever situações de erros prováveis e pistas para a descoberta de novos erros.
- *Inspection* – é a fase em que o objectivo é a efectiva procura de erros. No início é nomeado um moderador, que não deve coincidir com o autor,
- *Rework* – período onde são corrigidos os defeitos encontrados na fase anterior.
- *Follow-up* [19] – o moderador fica encarregue de assegurar que todas as questões foram fechadas e que todo o *rework* foi completado.

Para que este processo possa ser aplicado a pequenos projectos a equipa de projecto terá que ter no mínimo 3 elementos. Todavia, existem outros factores que podem eliminar, *a priori*, a possibilidade de este processo ser aplicado num determinado projecto:

- A gestão não acredita nesta metodologia,
- Ter que existir um moderador e este tem que ser treinado durante o processo,
- A inspecção requer uma estrutura de projecto modular,
- A inspecção é baseia-se na documentação.

Porém, devido à necessidade de treino e de especialização de elementos da equipa esta metodologia não serve os requisitos dos pequenos projectos. Adicionalmente, obriga a que seja produzida documentação em excesso, o que se poderia assumir como factor de atraso no caso dos pequenos projectos.

2.3.1.4. eXtreme Programming

A eXtreme Programming (XP) é uma metodologia exclusiva para desenvolvimento de projectos na área da engenharia de SW e é considerada uma das mais conceituadas metodologias de desenvolvimento ágil de SW [29] [11]. Esta metodologia nasceu a meio da década de 90 e teve como autor Kent Beck [12]. Na época, Beck estava envolvido num importante projecto na Chrysler do qual foi líder. Em 1999, este publicara o livro [12] onde explicava concretamente esta metodologia.

Segundo o livro de Beck, os objectivos do XP passam por:

- Conciliar produtividade com a qualidade do ser humano
- Criar um mecanismo de mudança social
- Assegurar um caminho para a melhoria
- Definir um estilo de desenvolvimento
- Construir uma disciplina de software

sendo o principal alvo do XP a redução dos custos associados à mudança.

Como qualquer metodologia, também a XP enfatiza um conjunto de valores [13]:

- *Communication*: É normalmente o *bottleneck* em projectos de SW devido à dificuldade de comunicação entre os vários *stakeholders* de um projecto.
- *Simplicity*: Cada pequeno aumento de complexidade num projecto implica um pequeno acréscimo de esforço mas um elevado aumento do risco. Esta metodologia tende a encontrar o caminho mais simples a seguir em qualquer situação. Requisitos devem ser simples, o programa deve ser simples e a disciplina em si deve ser simples. A simplicidade facilita a comunicação.
- *Testing*: Se a uma funcionalidade de um programa não está emparelhado um teste automático, então esta não funcionará. Se a funcionalidade tiver um teste automático, então é possível que esta funcione. XP contempla dois níveis de teste, os testes unitários e os testes funcionais. Testar fortifica a comunicação pois

implica uma dada sincronização entre os elementos da equipa de desenvolvimento. A comunicação incita a fase de testes pois exista um maior número de pessoas sensibilizadas para o mesmo problema. Testar aumenta a simplicidade pois, a partir dos testes, é possível tornar a estrutura do programa mais legível e a simplicidade, por sua vez, facilita a fase de testes dado que existe, naturalmente, menos a testar num programa mais simples.

- *Aggression*: Com os primeiros três valores postos em prática, a atitude necessária para se conseguir seguir esta metodologia é a de determinação e alguma “agressividade”. Em XP, um programador recomeça de novo o código quando este não está a correr bem e perde mais tempo em actividades de *refactoring* do que em actividades de desenvolvimento. Mais uma vez, todos os valores estão interligados. A comunicação potencia a “agressividade” pois duas pessoas têm mais coragem de testar algo mais difícil do que uma pessoa só. O contrário também se verifica na medida em que os programadores não estão retraídos a aprender novas áreas do sistema. Simplicidade suporta a “agressividade” por ser mais fácil manipular um programa mais simples. Reciprocamente, esta “agressividade” eleva a simplicidade pois daí advém uma maior entrega em descobrir a melhor maneira de estruturar um programa simples. Testar aumenta a “agressividade” pois é maior a motivação de utilizar algo já testado antes.

A conclusão mais valiosa que se pode tirar desta miscelânea de valores é a de que, no final é o factor humano o responsável por garantir o sucesso desta e de outras metodologias. Como tal, existem uma série de riscos relacionados com as práticas da metodologia em si e a aceitação dos elementos da equipa que apresentam probabilidades de acontecimento consideráveis. Estes risos são [32]:

- Problemas em trabalhar no cliente
- Oposição à metodologia
- Problemas com a tecnologia
- Falta de treino
- Programadores pouco dotados de *skills* necessárias

Não obstante, esta metodologia apresenta alguns pontos desalinhados com projectos onde o tempo é curto. O facto de se recomeçar o desenvolvimento de início em certos casos e despende de tempo para *refactoring*, não são as melhores medias a tomar quando o contexto é o de um pequeno projecto.

2.3.1.5. Rational Unified Process (RUP)

Rational Unified Process [8] [9] é um processo aplicado ao desenvolvimento de pequenos processos de software e que contempla a maioria das fases que compõem o ciclo de vida dos projectos [10]. É um processo iterativo de desenvolvimento composto por 4 fases:

- *Inception (What to build)*: âmbito do projecto, requisitos de alto nível, estimativa de recursos.
- *Elaboration (How to build it)*: detalhe nos requisitos, produção de uma arquitectura estável de acordo com casos de uso.
- *Construction (Build the product)*: completar os requisitos e desenhar o modelo, implementar e testar cada componente, recorrendo a protótipos nos utilizadores finais, lançar as primeiras versões do software funcional.
- *Transaction (Deploy to end users)*: produzir versões incrementalmente com base nos erros corrigidos, actualizar os manuais a cada versão,

Cada fase é composta por várias iterações [8] [33] representadas esquematicamente na Figura 4 em baixo:

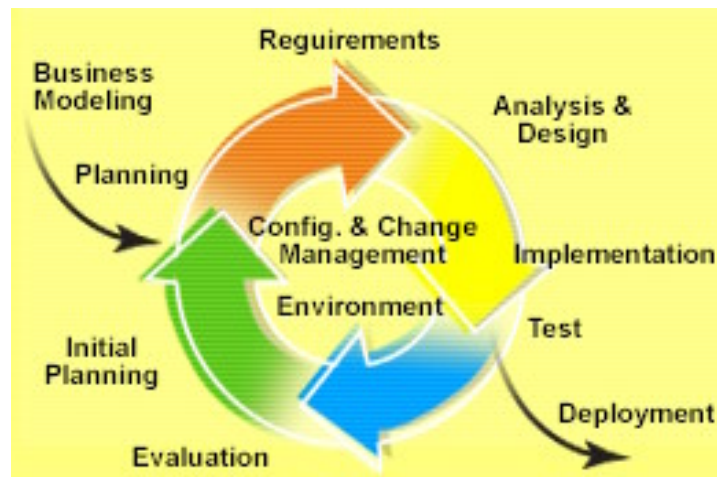


Figura 4. Fluxo típico de iterações da RUP (Fonte: [33])

A cada fase da RUP estão associadas fluxos da gestão comuns à maioria dos projectos. A ligação entre estas duas componentes bem como o peso de cada fluxo nas várias fases [8] encontram-se detalhados na Figura 5 em baixo:

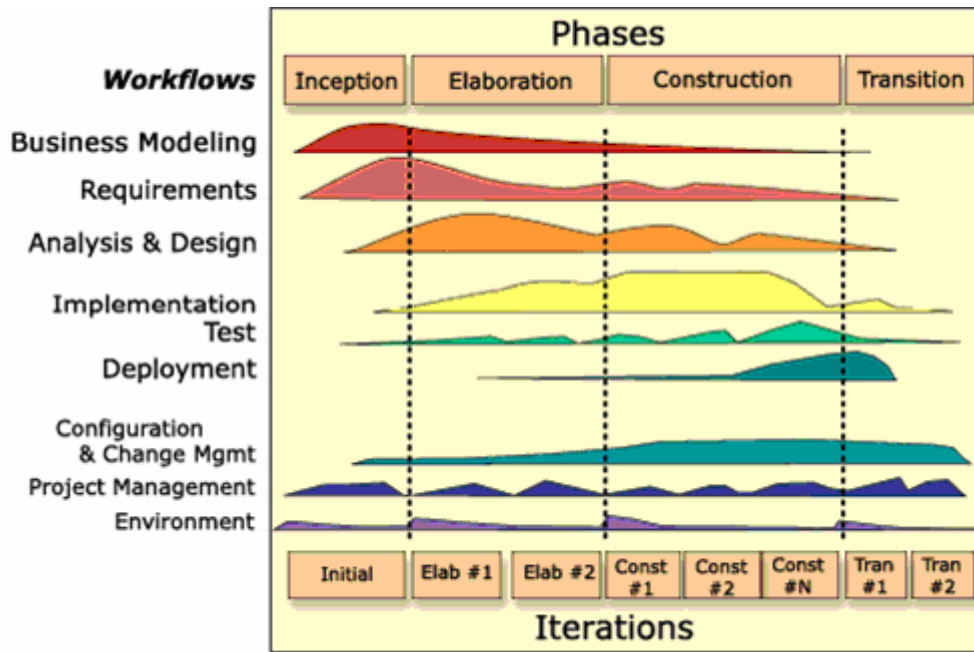


Figura 5. Diagrama geral da RUP (Fonte: [8])

Paralelamente a todas as fases e *workflows*, existe uma série de boas práticas e princípios que o modelo aconselha [34]:

- Desenvolver apenas o que é necessário (agilidade)
- Minimizar o trabalho em papel
- Ser flexível (requisitos, planeamento, etc)
- Aprender com erros antigos (*Feedback loops*)
- Estabelecer objectivos e critérios mensuráveis para avaliar o progresso

A orientação da RUP passa por dar menos ênfase à gestão de riscos do projecto. É visível que este sub-processo da gestão de pequenos projectos não se assume como uma actividade independente nos fluxos da metodologia, sendo por isso encapsulado noutra actividade de grão maior.

Devido á desvalorização atribuída à gestão de riscos e ao facto de esta metodologia ser mais focada e utilizada em projectos de desenvolvimento de software, pode concluir-se que esta metodologia não apresenta todas as características necessárias que a gestão de pequenos projectos procura.

2.3.1.6. PRINCE 2

PRINCE2 [35] [36] é uma metodologia de gestão de projectos estruturada. Segundo esta metodologia um projecto deve:

- Ter um arranque organizado e controlado de modo a que tudo seja planeado e devidamente organizado antes de se entrar no projecto.
- Garantir o controlo e organização durante a execução do projecto.
- Fazer uma análise no final do projecto de modo a que este seja, efectivamente, terminado.

A metodologia PRINCE2 consiste numa série de processos bem definidos que permitem orientar o gestor e a equipa de projecto. As actividades de controlo e organização num projecto são assumidas pelo gestor de projecto. Este tem a função de seleccionar as pessoas que irão na prática realizar o projecto e que, conseqüentemente, terão a responsabilidade de garantir que o mesmo é terminado no tempo estimado. Todas as actividades e condições de fecho do projecto, bem como as datas das mesmas, são ditadas na fase de arranque pelo gestor de projecto.

Cada projecto que segue PRINCE2 contempla um quadro de projecto feito pelo cliente. O gestor de projecto fica incumbido de reportar, regularmente, informação actualizada acerca do progresso do projecto ou de algum problema pertinente para o quadro de projecto. Por outro lado, o quadro de projecto deverá fornecer ao gestor de projecto as decisões necessárias para que o projecto possa continuar, ultrapassando todas as dificuldades.

No PRINCE2, existem três vistas distintas de garantia: negócio, utilizador e especialista. Cada uma reflecte o interesse dos vários membros do quadro de projecto. Esta função passa por avaliar em cada fase a viabilidade do projecto ao nível da dinâmica custos/benefício (ponto de vista do negócio), verificar se os requisitos definidos pelos utilizadores estão a ser cumpridos e, do ponto de vista técnico, se o projecto cumpre os níveis de qualidade especificados. Esta tarefa pode ser delegada a uma equipa externa ao projecto chamada *Project Assurance Team* de modo a tornar este processo o mais coerente e neutro possível.

É comum na maioria dos projectos haverem bastantes tarefas administrativas, como por exemplo, marcação de reuniões, gerir ficheiros ou manter os recursos informados. Normalmente, são os gestores de projecto que assumem este papel, principalmente em pequenos projectos. No entanto, em ambiente multi-projecto, torna-se algo complicado para o gestor dos projectos gerir os mesmos e administrar todos eles. Em algumas empresas atribuem-se as tarefas administrativas a uma equipa paralela para ajudar o gestor de projecto.

Para além desta metodologia definir com algum rigor toda a componente de recursos humanos que participam num projecto e suas responsabilidades, ela também tem uma forte componente de gestão de risco, gestão de qualidade e gestão de mudança.

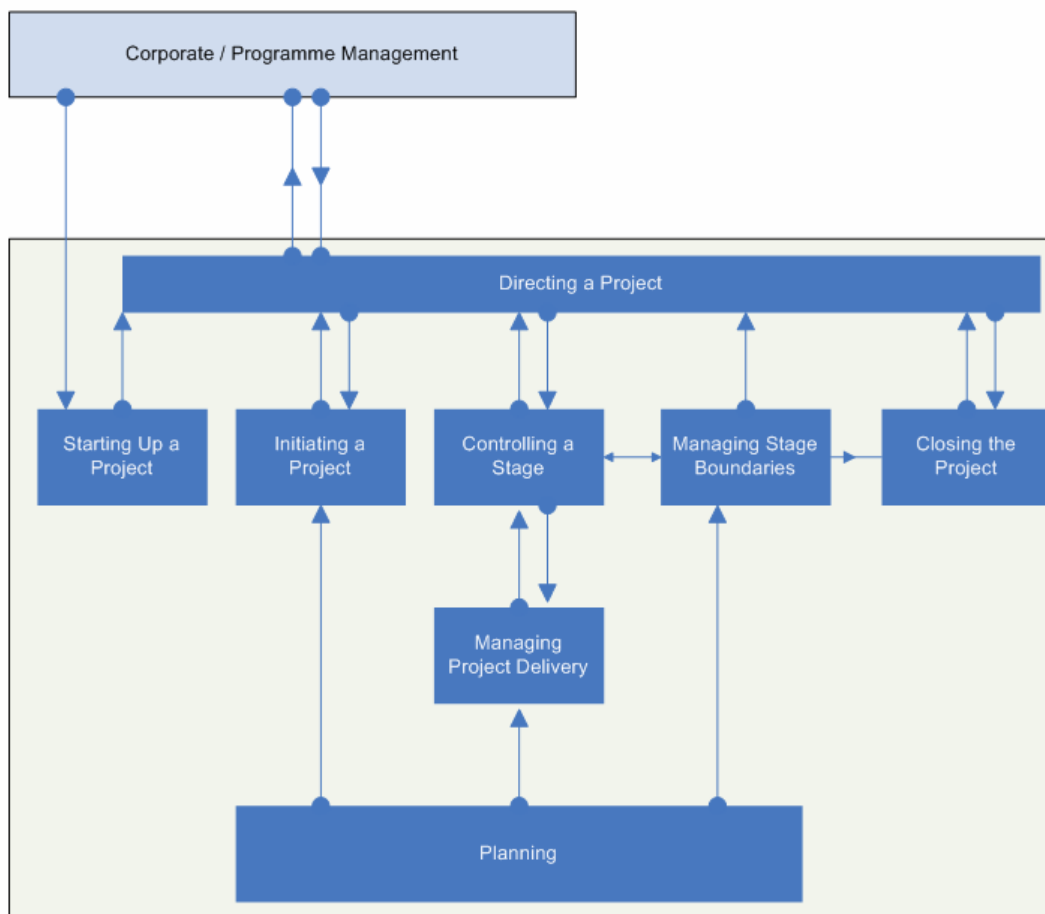


Figura 6. Processos que constituem a metodologia PRINCE2 (baseado em [35] e [36])

À gestão de risco está relacionada com a avaliação do que pode correr mal num projecto, com as consequências dos riscos identificados e com os métodos de recuperação dos mesmos. A gestão da qualidade garante a verificação das especificações definidas com o cliente através de testes e revisões. A componente de gestão de mudança está relacionada com as mudanças de requisitos propostas pelos clientes ou outros factores do próprio projecto e da sua implementação.

Contudo, a gestão de projectos é uma disciplina complexa e é errado assumir que aplicando apenas o PRINCE2 a um projecto, este tenha sucesso garantido, sendo que existem projectos onde alguns dos processos da metodologia não são sequer aplicáveis.

Como foi referido e ilustra a Figura 6, o PRINCE2 é constituído por um conjunto de processos, mais exactamente 8 (agrupando 45 sub-processos), cada um deles direccionados para uma fase distinta do projecto:

- Arranque (*Starting up a Project*) – escolhe-se a equipa de projecto e sumariza-se a abordagem geral mesmo. Só se inicia a fase seguinte se o quadro de projecto assim o decidir.

- Planeamento (*Planning*) – estimar-se o esforço requerido pelas actividades que dão origem ao produto e calendarizam-se todas as actividades num plano. Nesta fase deve fazer-se a análise de riscos e os critérios de fim de cada fase (*milestone*) e do projecto como um todo.
- Inicializar o projecto (*Initiating a Project*) – definem-se os critérios de qualidade e controlo do projecto. Distribui-se o plano de projecto previamente aprovado pelo quadro de projecto.
- Gerir o projecto (*Directing a Project*) – definem-se as responsabilidades do quadro de projecto e o modo como este deve efectuar o controlo do mesmo. Adicionalmente, é detalhado o modo como o planeamento deve ser feito no caso de acontecerem atrasos ou outros imprevistos.
- Controlar uma fase (*Controlling a Stage*) – PRINCE2 sugere que os projectos sejam divididos em fases e os sub-processos que propõem sejam mapeados nessas fases de modo a que estas possam ser correctamente controladas. São especificados os modos como o progresso deve ser monitorizado e como deve ser reportado ao quadro de projecto. É discutido em conjunto um modo de encontrar questões importantes sobre o projecto e o processo de tomada de medidas para os resolver.
- Gerir a entrega do produto (*Managing Product Delivery*) – diz respeito à aceitação, execução e entrega de um pacote de trabalho.
- Gerir o âmbito de uma fase (*Managing Stage Boundaries*) – o que deve ser feito no fim de cada fase. Planeamento, gestão de riscos e de alterações devidamente melhoradas de acordo com o impacto desta fase e, por fim, como o fim desta fase deve ser reportado.
- Encerramento (*Closing the Project*) – este processo cobre todas as tarefas que devem ser feitas no final do projecto. É necessário libertar os recursos e avaliar o projecto.

2.3.1.7. Processos de desenvolvimento projectos de software

Para além das metodologias referidas, existem algumas propostas vindas de outros autores [37] [38] [39] [40] [41] [42] menos praticadas mas não menos interessantes.

A proposta apresentada por Russ e McGregor [37], tem como objectivo capturar todas as actividades essenciais no desenvolvimento de produtos de software e definir um processo com base nestas actividades.

Este processo tem em conta uma característica bastante presente nos pequenos projectos que é o facto de estes serem iterativos e incrementais.

Partindo deste ponto, o processo conta ainda com técnicas de inspecção e métricas. As técnicas de inspecção, de um modo geral, focam-se em examinar os produtos, neste caso de software, para encontrar erros ou defeitos, com suporte em casos de teste feitos especialmente para aquele produto.

As métricas servem para ajudar os elementos da equipa de desenvolvimento a analisar e melhorar a sua própria produtividade e qualidade. Mas, segundo a perspectiva desta metodologia, esta última técnica por si só não é suficiente, e, como tal, para além das métricas existe um outro processo que guia cada elemento da equipa nos passos que há-de dar e qual o esforço e ênfase a atribuir a cada actividade em cada fase.

Este processo visa eliminar o excesso de documentação requerida pela gestão no processo de desenvolvimento, reconfigurar as tarefas de modo a otimizar o fluxo das mesmas, e eliminar algumas tarefas menos importantes ou integrá-las noutras.

Buehring [38] defende a ideia de que, num projecto, o que mais interessa é cumprir com os objectivos de negócio e tudo o que não esteja no seguimento desta ideia pode ser perfeitamente eliminado, em especial a documentação que tende a ser bastante nos pequenos projectos. Ora, segundo o autor, pode partir-se da seguinte premissa: “Se nos ajudar a cumprir com os objectivos de negócio, então produz-se, caso contrário, não vale a pena perder tempo” [38].

Esta metodologia passa por seguir um processo onde estão incluídas as actividades básicas da gestão de projectos:

- Definir âmbito e objectivos do projecto
- Definir as entregas do projecto
- Planeamento do projecto consoante as datas definidas para as entregas
- Comunicação
- Controlo e *Reporting*
- Gestão de risco

Por fim, Dietrich et al [39] [40], Bussler [41] e Engwalla pretendem certificar-se de que o síndrome da alocação de recursos em múltiplos projectos é a principal dificuldade enfrentada pelos gestores neste ambiente. Para tal, efectuaram estudos em várias empresas distintas e concluíram que este é um factor comum. O problema surge, entre outros motivos, pelo facto das aplicações de suporte em muitos casos não estarem preparadas para este tipo de gestão. Esta área está ainda aberta a investigação e as respostas para este tipo de problemas está ainda em aberto.

2.3.1.8. Scrum

O termo *Scrum* é oriundo da estratégia utilizada em rugby que corresponde à recuperação de uma bola perdida de novo para jogo, devido ao trabalho de equipa [43].

No contexto da gestão de projectos, o *Scrum* é um processo iterativo e incremental para desenvolver e gerir qualquer tipo de projecto ou produto. No fim de cada iteração é produzido um conjunto de novas funcionalidades. Este processo tem como principal foco a forma como os elementos da equipa devem conduzir o seu trabalho de modo a produzir um resultado flexível num ambiente de constante mudança e não impõem qualquer método específico ou práticas de desenvolvimento de SW. Em vez disso, o *Scrum* requer certas ferramentas e práticas de gestão em diferentes fases do processo de modo a evitar o imprevisível e o complexo [44].

As práticas chave deste processo estão descritas de seguida e representadas na Figura 7 [11]:

- *Product Backlog* – É a lista de prioridades de todas as funcionalidades e alterações ao produto propostas pelos vários actores (cliente ou equipa de projecto). O dono do produto é o responsável por manter o *Product Backlog*.
- *Sprints* – É um período de 30 dias onde são definidas as várias variáveis do projecto, tais como requisitos, tempo, recursos e tecnologia. As ferramentas de trabalho da equipa são o *Sprint Planning Meetings*, *Sprint Backlog* e o *Daily Scrum Meetings*.
 - *Sprint Planning Meetings* – É a fase onde se definem os objectivos e funcionalidades e onde participam o cliente, a gestão e os utilizadores finais. De seguida, O *Scrum Master* (gestor de projecto) e a *Scrum Team* (restantes elementos da equipa de projecto) determinam como o produto vai ser implementado durante a *Sprint*.
 - *Sprint Backlog* – É a lista de funcionalidades que está actualmente ligada a uma *Sprint* em particular. Quando todas as funcionalidades estiverem implementadas uma nova iteração do sistema é entregue.
 - *Daily Scrum* – É a reunião diária de, aproximadamente, 15 minutos cujo objectivo é o de reportar o progresso do projecto e superar os obstáculos encontrados por todos os elementos da *Scrum Team*.

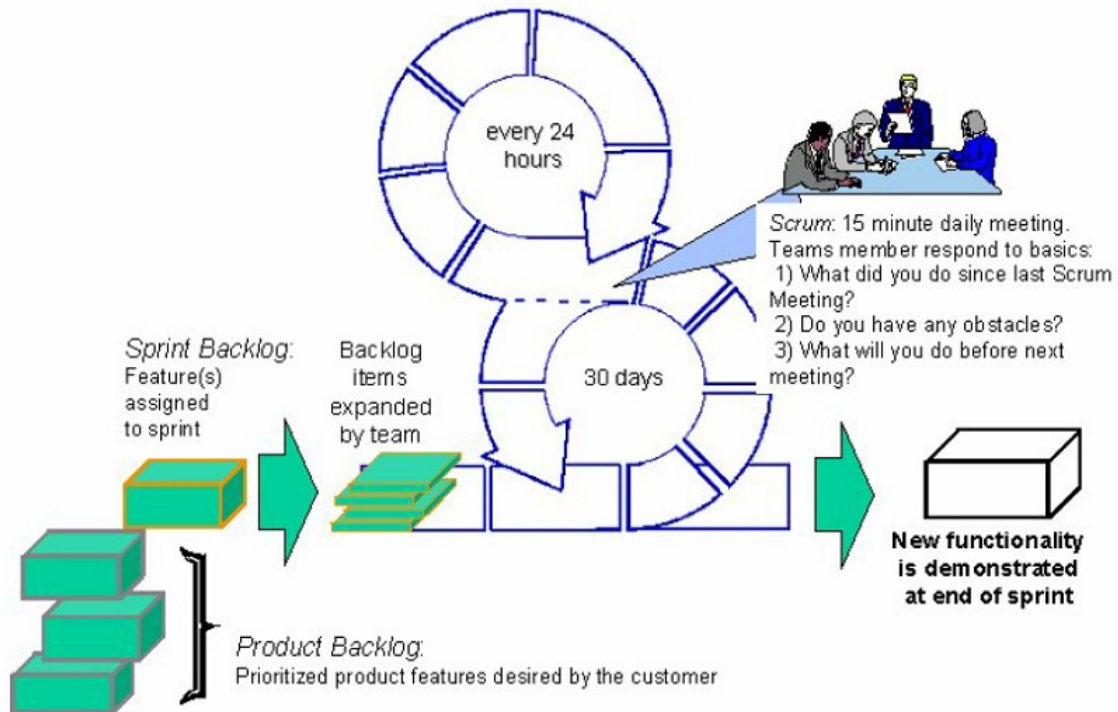


Figura 7. Processo da Scrum [11]

O Processo *Scrum* tem impactos consideráveis nos métodos de trabalho dentro de uma organização. O papel do gestor de projecto (*Scrum Master*) deixa de ser o de gerir uma equipa, pois cada elemento da equipa se organiza a si próprio e toma as decisões que vão aparecendo. *Ken Schwaber* diz que: “A maioria das funções da gestão são utilizadas no sentido de dirigir um projecto, dizer à equipa o que fazer e depois assegurar o seu cumprimento. *Scrum* aponta para organizações onde as equipas decidem o que fazer enquanto a gestão assume o papel de retirar as barreiras do caminho” [45].

Rising e Janof [44] sugerem: “Claramente, *Scrum* não é uma abordagem para equipas grandes e estruturas complexas, mas nós descobrimos que até equipas pequenas e isoladas num projecto grande, conseguem tirar partido de alguns elementos do *Scrum*”. Contudo, é ainda necessário um esforço maior e mais casos práticos para suportar esta afirmação.

2.3.1.9. Feature Driven Development (FDD)

Feature Driven Development foi utilizado pela primeira vez no final dos anos 90 e, ao contrário das restantes metodologias, esta está mais focada nas fases de desenho e implementação de software [46].

É visível pela Figura 8 que esta metodologia está dividida em 5 fases. As primeiras três são efectuadas no início do projecto, enquanto as restantes duas correspondem ao ciclo iterativo do processo e suportam o desenvolvimento ágil com rápidas adaptações e alterações aos

requisitos e às necessidades de negócio. O FDD pressupõe entregas frequentes sempre sincronizadas com monitorização e *reporting* do progresso [11].

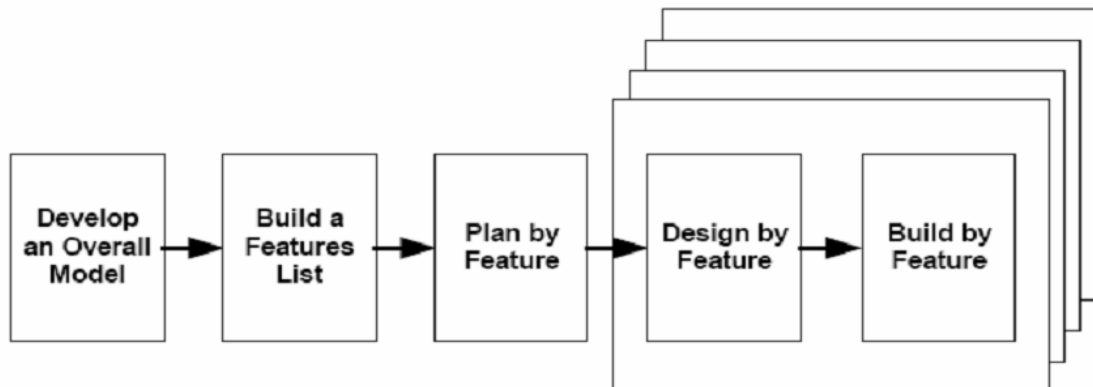


Figura 8. Fases do Processo do *Feature Driven Development* [46]

- *Develop an Overall Model* – É definido o âmbito do projecto e seu contexto, sendo este apresentado a toda a equipa. São levantados os requisitos, casos de uso e funcionalidades do projecto.
- *Build a Features List* – É criada uma lista de características de modo a suportar os requisitos.
- *Plan by Feature* – A equipa de desenvolvimento ordena os conjuntos de características e funcionalidades por prioridades e dependências e aloca-as aos programadores seniores. Além disso, são identificados os *milestones* do projecto e, conseqüentemente, as calendarizações do mesmo.
- *Design by Feature e Build by Feature* – São seleccionadas funcionalidades e atribuídas às equipas responsáveis pelo seu desenvolvimento. O sub-processo seguinte passa pela produção dos diagramas de sequência para as funcionalidades seleccionadas, que são passados aos programadores que, por sua vez, as implementam de acordo com as necessidades de desenho de cada funcionalidade específica. Nesta fase do processo podem existir várias equipas a desenhar e implementar concorrentemente outras funcionalidades. Por fim, todo o código desenvolvido é testado e inspeccionado, sendo que em caso de sucesso, as funcionalidades são adicionadas à implementação principal.

Esta metodologia é bastante orientada ao trabalho colaborativo e é fácil de por em prática. Porém, é muito restrita ao desenvolvimento de SW e contempla uma fase de inspecção que, como já foi referido no subcapítulo 2.3.1.3, apresenta alguns problemas.

2.3.1.10. Dynamic System Development Method (DSDM)

Esta metodologia foi desenvolvida na década de 90 no Reino Unido e define-se como uma mistura de desenvolvimento rápido de SW e práticas de desenvolvimento iterativas [47]. Martin Fowler [48], um dos escritores do *Agile Manifesto*, acredita que “DSDM é notável por reunir muita da infra-estrutura das várias metodologias tradicionais de maturidade, enquanto segue os princípios das abordagens dos métodos ágeis”. A ideia fundamental desta metodologia é de fixar tempo e recursos e ajustar o número de funcionalidades de acordo com estes limites. Este processo consiste em 5 fases [11] [49]:

- *Feasibility Study* – Nesta fase decide-se se é ou não apropriado utilizar DSDM ou não tendo em conta o tipo de projecto, o tipo de organização e o método de trabalho.
- *Business Study* – Para esta fase, recomenda-se a organização de um workshop para ajudar a perceber o domínio de negócio do projecto. Os resultados chave desta fase são a arquitectura do sistema e um protótipo do mesmo.
- *Functional Model Iteration* – Primeira fase iterativa e envolve análise, codificação e prototipagem. Os ganhos da prototipagem servem como input da análise e a iteração pára quando o modelo funcional estiver em equilíbrio.
- *Design and Built Iteration* – O sistema é construído maioritariamente nesta fase. Os protótipos da fase anterior são revistos pelos utilizadores e a implementação seguinte é baseada nos seus comentários.
- *Implementation* – Nesta fase encontram-se as tarefas de concretização e instalação do produto como a formação, elaboração dos manuais e a revisão final do projecto. No entanto, se algo se manifestar como errado nesta fase então o projecto poderá integrar novamente qualquer uma das fases anteriores de modo a serem efectuadas as correcções necessárias.

O DSDM é um processo que requer interacção com os utilizadores, entregas frequentes, equipas motivadas e fases de teste durante o processo. Todavia, é sabido que qualquer um destes requisitos é difícil de garantir individualmente e ainda mais como um todo, o que leva a uma falta de confiança na sua aplicação principalmente em projectos de pequena dimensão.

2.3.1.11. Adaptive Software Development (ASD)

Adaptive Software Development foi desenvolvida por *James A. Highsmith* e não é uma metodologia de desenvolvimento de SW, mas antes uma aproximação ou uma atitude que deve ser adoptada quando uma organização aplica processos ágeis [50] [11]. A base desta metodologia é a substituição da tradicional fase de planeamento estática por um ciclo dinâmico que é composto essencialmente por 3 premissas, especular, colaborar e aprender.

Highsmith vê o planeamento como um paradoxo num ambiente adaptativo. Normalmente, no modo de planear típico, quando algo não é planeado é visto como um erro. Num ambiente adaptativo estas situações são aquelas que, muitas das vezes, guiam à solução.

Como a maioria das restantes metodologias apresentadas, também a ASD se divide em fases, neste caso 3 fases não lineares e concorrentes:

Speculate – Define-se a missão do projecto e esclarecem-se os pontos mais instáveis.

Collaborate – Enaltece-se a importância do trabalho de equipa no desenvolvimento de sistemas de constante mudança.

Learn – Sublinha-se a necessidade de admitir e reagir a erros e que os requisitos se alteram durante o desenvolvimento.

ASD foca-se mais nos resultados e na sua qualidade do que nos processos usados para produzir esses resultados. Num ambiente imprevisível as pessoas devem estar preparadas para enfrentar o incerto e necessitam de colaborar nesse sentido. A gestão é entendida como o elemento encorajador da comunicação e criatividade em vez de limitar as pessoas a regras sobre o que fazer.

Nos modelos tradicionais, os métodos de trabalho seguem sempre uma receita e a capacidade de aprender é limitada. *Highsmith* defende que, “Num ambiente adaptativo, aprender desafia todos os *stakeholders* a reflectir sobre as suas ideias e a usar esses resultados para aplicar nos próximos projectos” [47] [50]. Ora, aprender é um processo contínuo que assume que o planeamento e o desenho devem evoluir consoante o progresso do desenvolvimento.

2.3.1.12. Conclusão

O gráfico presente na Figura 9, ilustra a relação entre as variáveis ‘Facilidade de Utilização’, ‘Especificidade’, ‘Período de Implementação’ (área dos círculos) e ‘Necessidade de recursos’ (cor dos círculos) para as metodologias descritas.

Por um lado, existem metodologias bastante específicas para um determinado tipo de projectos, p. ex. CMM, DSDM, RUP, XP, FDD e Scrum (que foram desenhadas para projectos de desenvolvimento de SW), e, como tal, impossibilitam ou dificultam a sua aplicação a outro tipo de projectos.

Por outro lado existem metodologias mais generalistas, p. ex. PRINCE2, Virtual Members e Inspection Method, mas que se apresentam igualmente bastante complexas e demoradas de implementar. Estes dois factores são incompatíveis com os pequenos projectos, bem como a elevada necessidade de recursos para por estas metodologias em prática.

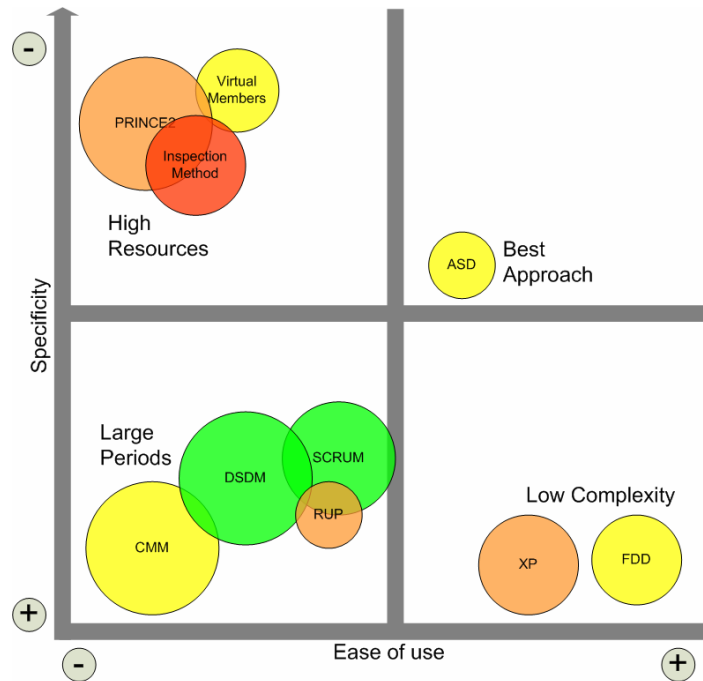


Figura 9. Análise das Metodologias (Área do círculo – período de implementação; Cor do círculo – recursos necessários)

Como conclusão deste gráfico fica a inexistência de uma metodologia que, no seu conjunto, reúna os requisitos dos pequenos projectos: fácil de utilizar, aplicável a qualquer tipo de projecto e possível de implementar em pouco tempo com poucos recursos.

2.3.2. Ferramentas

Na actualidade é assumido que qualquer projecto tem que ser gerido com o suporte de uma ferramenta apropriada. Quando esses projectos são pequenos, poucas são as ferramentas que desempenham bem esse papel. Para projectos de pequena escala as ferramentas de suporte devem obedecer a certos requisitos que a maioria das ferramentas que existem para grandes projectos não apresenta.

Uma ferramenta de apoio à gestão de projectos em geral deve ser capaz de [20]:

- Fornecer a informação certa,
- No formato correcto,
- Às pessoas certas,
- No tempo certo.

2.3.2.1. Microsoft Project

De todas as ferramentas de suporte à gestão de projectos, a mais utilizada é o Microsoft Project (MS Project).

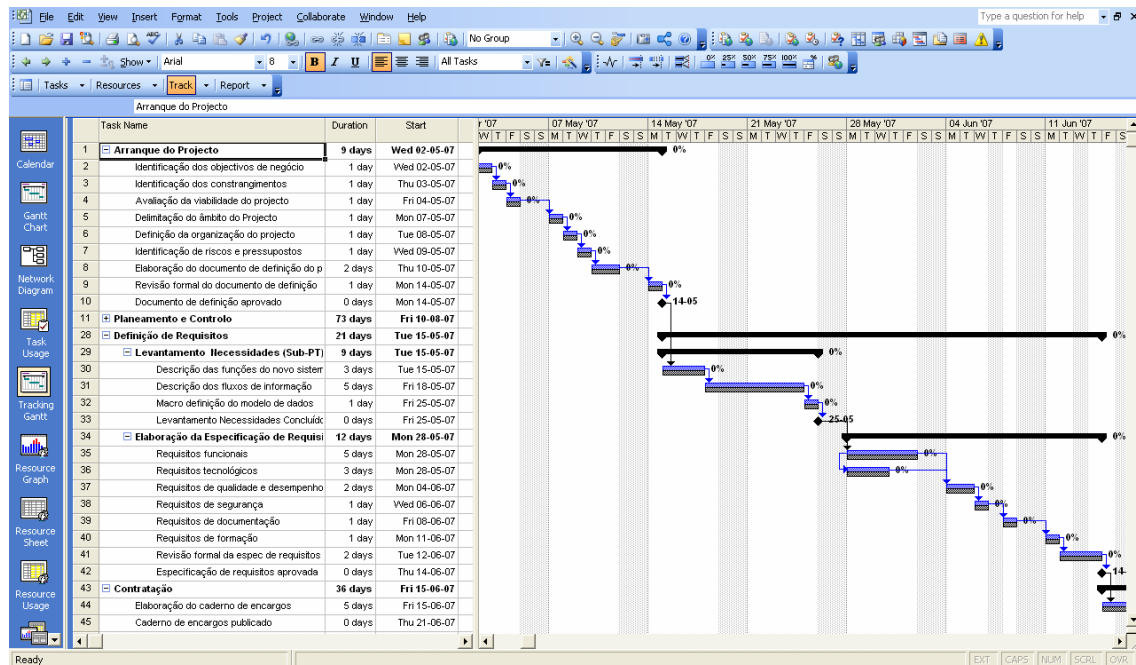


Figura 10. Microsoft Project

Como mostra a Figura 10, esta ferramenta na sua generalidade desempenha bem o seu papel ao nível das funcionalidades de planeamento quando os projectos em questão se tratam de grandes projectos, mas falha em bastantes pontos na perspectiva dos pequenos projectos [20] [21] [22] [23].

Alguns estudos (Tabela 2) mostram que esta ferramenta não é fácil de aprender e utilizar sem treino [24].

Tabela 2. Satisfação com ferramentas de gestão de projectos [24] (escala 0-5)

Ferramenta	Facilidade de Utilização
Project Scheduler	3.2
Primavera Project Planner	3.2
Project Workbench	3.5
Microsoft Excel	3.8
Primavera SureTrak	3.2
CA-SuperProject	3.0
Microsoft Project	3.6
Artemis Prestige	2.0
FasTracs	3.4
Time Line	3.2

Contudo, a maior falha do MS Project passa pela ausência de mecanismos de colaboração e comunicação. Não existem fóruns de discussão, salas de *chat*, apresentações, notas ou hyperlinks, nada que apoie os elementos da equipa de projecto para comunicarem.

Como consequência, a gestão de riscos acaba por não ser suportada pela ferramenta e na óptica dos pequenos projectos este é um factor de extrema importância, dado o ambiente de mudança a que estes estão sujeitos.

2.3.2.2. Quickbase

É uma ferramenta bastante completa do ponto de vista do planeamento de um projecto. Apresenta as funcionalidades básicas de calendarização, gestão de recursos e gestão de riscos mas, analogamente ao MS Project, falha no suporte à comunicação entre a equipa.

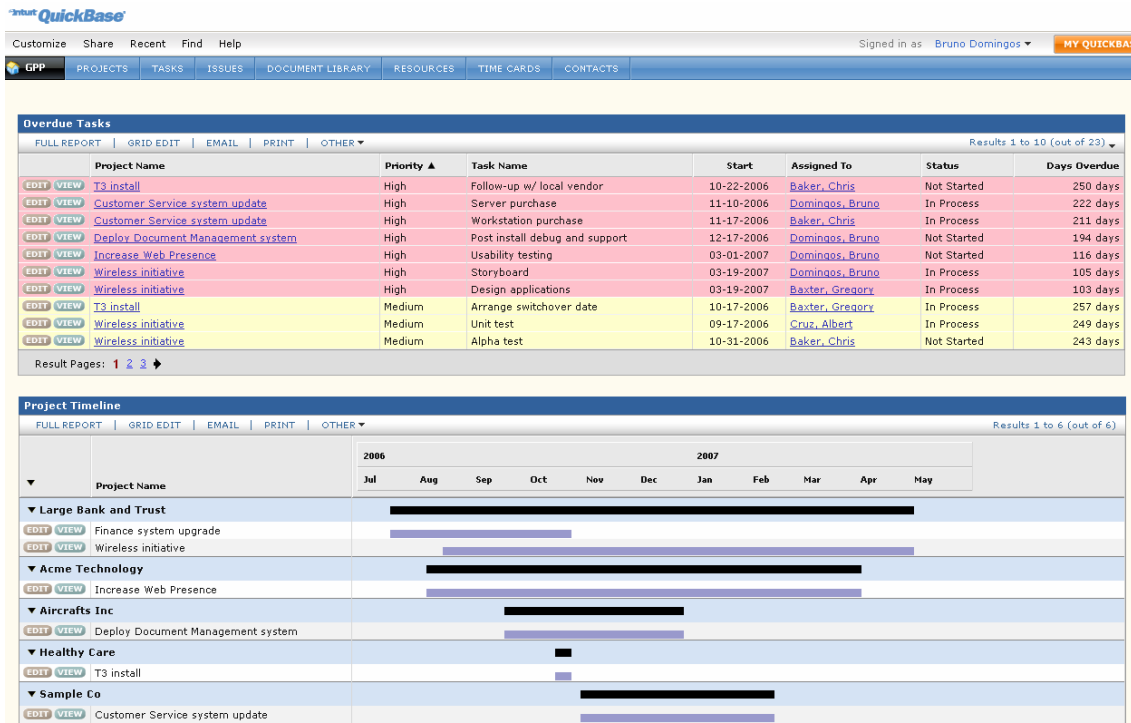


Figura 11. Quickbase

O Quickbase apresenta-se como uma aplicação Web, o que garantiria, a priori, facilidade de utilização. Como se pode visualizar na Figura 11, tal facto não se verifica, visto que a ferramenta apresenta demasiada informação simultânea para o utilizador. Não é fácil encontrar a informação pretendida sem que antes se tenham que seleccionar vários critérios de filtragem. Não é, portanto, uma ferramenta que dispense treino e especialização por parte de quem a utiliza.

2.3.2.3. Basecamp

Ao contrário das ferramentas anteriores, a Basecamp (Figura 12), apresenta uma forte componente de suporte à comunicação. Existem salas de *chat*, mensagens, fóruns de discussão e notas. É muito fácil utilizar esta ferramenta nesta perspectiva.



Figura 12. Basecamp

Por outro lado, esta ferramenta é muito incompleta em todos os outros campos. Ao nível do planeamento e controlo apenas apresenta como funcionalidades as conhecidas *to-do lists* e marcação de *milestones*. Não existe o conceito de tarefa e, conseqüentemente, é impossível criar dependências entre estas ou alocá-las por vários recursos em termos de esforço.

A gestão de riscos foi também completamente esquecida. Não existe maneira de controlar os riscos e definir responsabilidades a partir da aplicação.

2.3.2.4. Conclusão

O gráfico presente na Figura XXX, ilustra a relação entre as variáveis 'Facilidade de Utilização', 'Comunicação', 'funcionalidades de controlo e planeamento' (área dos círculos) e 'Funcionalidades de gestão de recursos' (cor dos círculos) para as ferramentas analisadas.

O MS Project apresenta-se como uma ferramenta bastante completa a nível das funcionalidades de planeamento, controlo e gestão de recursos mas, falha visivelmente nos mecanismos de suporte a ambientes colaborativos e facilidade de utilização.

Muito próximo do MS Project encontra-se a ferramenta Quickbase que é, ainda assim mais fraca a nível das referidas funcionalidades.

Do lado oposto encontra-se a ferramenta Basecamp que é praticamente desprovida de funcionalidades de planeamento, controlo e gestão de recursos. No entanto esta ferramenta é muito completa relativamente a mecanismos de comunicação e colaboração entre a equipa.

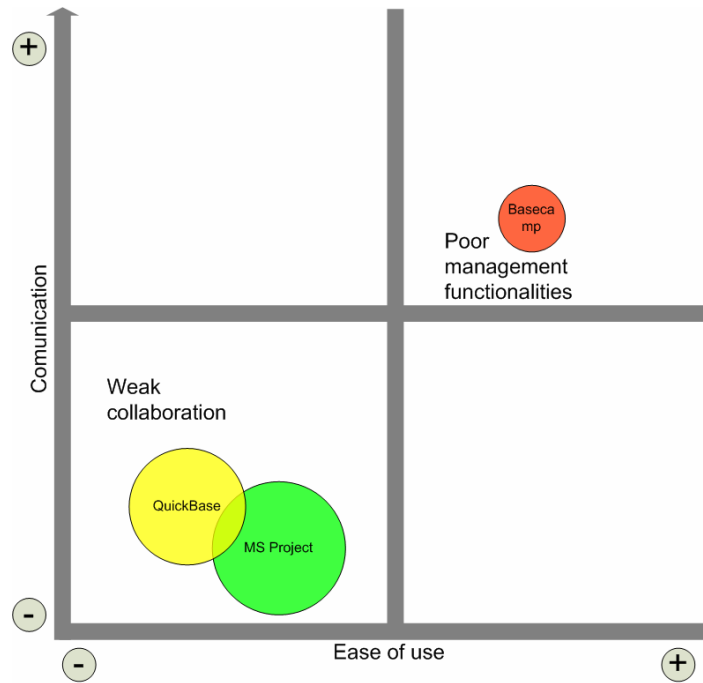


Figura 13. Análise das Ferramentas (Área do círculo – Funcionalidades da planeamento e controlo; Cor do círculo – Funcionalidades de gestão de recursos)

Tabela 3. Comparação entre ferramentas

	MS Project	Quickbase	Basecamp
Facilidade de Utilização	●●●	●●	●●●●●
Não necessita de treino/especialização	●	●●	●●●●●
Funcionalidades de planeamento e controlo	●●●●●	●●●●●	●●
Gestão de recursos	●●●●	●●●●	●
Gestão de riscos	●●	●●●	●
Gestão de requisitos	●	●	●
Espaço para documentação	●	●●●●	●
Suporte à comunicação	●	●	●●●●●

(● – Inexistente; ●● – Fraco; ●●● – Médio; ●●●● – Bom; ●●●●● – Muito Bom)

Na Tabela 3 foram classificadas estas ferramentas tendo em conta as características essenciais de ferramentas de apoio à gestão de pequenos projectos [21] [22].

É visível que estas ferramentas apresentam problemas. Os pontos avaliados são de extrema importância numa ferramenta de apoio à gestão de pequenos projectos e nos casos avaliados apresentam-se por vezes como fracos ou mesmo inexistentes.

Para concluir, não se pode ignorar uma premissa básica que todos os gestores devem ter em mente: “Um computador não consegue gerir um projecto. Este apenas ajuda as pessoas a desempenhar melhor os seus papéis.” [20].

3. Pequenos Projectos

3.1. Projectos de Sistemas de Informação

A gestão de projectos é uma prática já antiga no ambiente empresarial. No entanto, não quer isto dizer que esta seja uma área desprovida de problemas ou falhas, pelo contrário, esta disciplina apresenta uma taxa de sucesso bastante reduzida, especialmente na área projectos de SI/IT [51]. Há anos 50 atrás *Oisen* [52] referiu que os critérios de sucesso de um projecto diziam respeito, unicamente, a três variáveis: qualidade, tempo e custos. Mais tarde, *Wright* [53] reduziu este universo apenas em duas variáveis: Tempo e Orçamento. Muitos outros autores, *Turner* [54], *Morris e Hough* [55], *Wateridge* [56], de *Wit* [57], *McCoy* [58], *Pinto e Slevin* [59], *Saarinen* [60] e *Ballantine* [61] vieram a concordar com *Oisen* [52], aceitando a dinâmica das três variáveis, mas não exclusivamente. Como mostra a Figura 14, são, nos dias de hoje, muito mais do que simplesmente estes três factores que ditam o sucesso de um projecto.

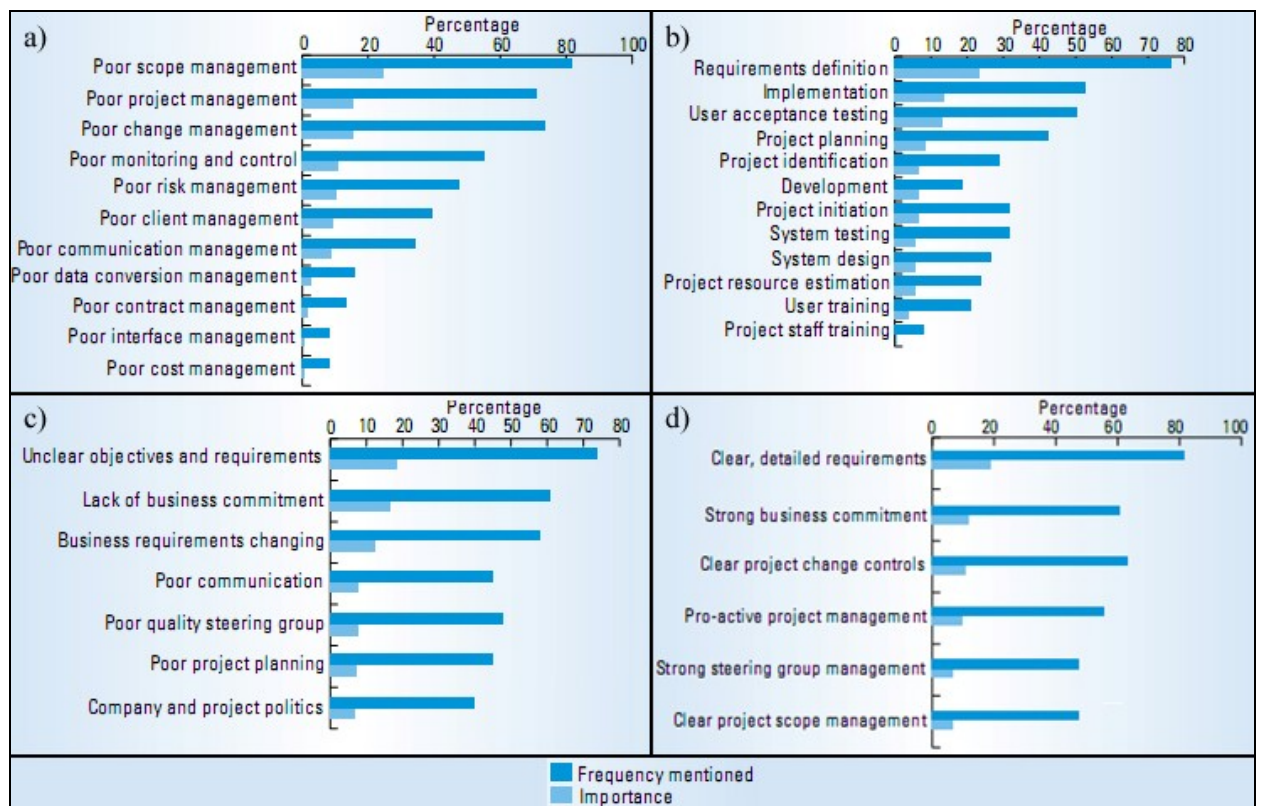


Figura 14. a) Actividades de gestão que contribuem para o insucesso dos projectos de TI (Fonte: [62]); b) Fases de insucesso (Fonte: [62]); c) Causas de insucesso (Fonte: [62]); d) Factores críticos de sucesso (Fonte: [62])

“Mais de 250 biliões de dólares são gastos nos E.U.A. todos os anos em, aproximadamente, 175,000 projectos de sistemas de informação. Apenas 26% destes projectos são terminados nos tempos previstos e dentro dos orçamentos.” [63].

Um estudo feito pelo *Standish Group* publicado na revista *PC Week*, revelou que “31% dos projectos de SI são cancelados antes de serem completados a um custo estimado combinado de 81 biliões de dólares. Além disso, 52,7% dos projectos que são completados custam 189% do orçamento inicial, o que se traduz em 59 biliões de dólares.” [64]. Números mesmo assustadores, que conduzem a uma situação de crise na indústria das TI.

Existem vários estudos sobre esta problemática que, inevitavelmente, se tem vindo a agravar ao longo dos anos. Segundo alguns depoimentos de gestores de projecto e outros especialistas, e, por análise dos dados retirados da Figura 14, pode afirmar-se que a origem dos sucessivos fracassos dos projectos de SI e TI se devem à combinação de diferentes factores como [64] [27] [2] [65] [66]:

- Âmbito do projecto mal definido
- Falta de acordo na definição de um conjunto bem articulado de objectivos do projecto
- Falta de dedicação às tarefas de gestão de projecto que na maioria das vezes são delegadas a elementos mais técnicos que não apresentam capacidades para tomar decisões críticas mais direccionadas para os gestores seniores.
- Gestores ignoram as boas práticas e as experiências passadas
- Fraca gestão, planeamento e controlo devido à falta de sistemas para medir o progresso e identificar potenciais riscos a tempo de reduzir os seus impactos.
- Fixação nas estimativas iniciais
- Alterações ao projecto são geridas de forma desatenta
- Estimativas irrealistas
- Nível de detalhe errado
- Falta de conhecimento acerca das ferramentas de gestão de projecto
- Demasiada confiança no software de gestão de projectos
- Demasiados recursos
- Elementos da equipa de projecto não apresentam as características técnicas necessárias
- Problemas entre os elementos da equipa
- Recompensa pelas acções erradas
- Requisitos mal definidos
- Gestores de projecto não entendem as necessidades dos clientes
- Problemas de qualidade descontrolados
- Adopção de novas tecnologias
- Tecnologia escolhida muda ou não é apropriada para suportar o projecto

- Necessidades de negócio mudam
- Utilizadores dos sistemas são cépticos e resistentes à mudança
- *Sponsor* do projecto vai-se afastando.

Os problemas mencionados estão sobretudo relacionados com más práticas de gestão e pouca dedicação aos projectos em causa. Consequentemente, nenhuma destas causas pode ser resolvida com base numa ferramenta ou metodologia mas antes através de uma melhor formação e gestão de recursos porque, na maioria das vezes, os gestores de projecto, devido ao elevado número de projectos ou outras tarefas que têm em mãos, não dispõem do tempo necessário para executar todo o seu trabalho com qualidade. A formação é igualmente um problema das empresas, pois são estas as responsáveis por formar os seus colaboradores.

3.2. Gestão de Pequenos Projectos

Quando se fala de gestão de projectos, raramente se faz a devida dissociação entre os chamados projectos de pequena e grande escala. No entanto, existem algumas características que fazem realmente a diferença quando é necessário planear um pequeno projecto e, ao mesmo tempo, definir uma boa plataforma de suporte ao mesmo.

A definição de um pequeno projecto não é uma definição estanque. Para alguns especialistas um pequeno projecto varia entre uma pessoa durante um mês até cinco pessoas durante seis meses [20] [67]. Para outros, um pequeno projecto varia de uma pessoa durante um dia até seis pessoas durante seis meses [25] [26]. Com base na literatura e atendendo à realidade das empresas em Portugal, estes classificam-se como projectos que envolvem desde uma pessoa / três dias a seis pessoas / dois meses. Pode-se também enquadrar um pequeno projecto entre valores monetários, mas este é um parâmetro que difere de organização para organização de acordo com a sua própria dimensão.

3.2.1. Gestor de Projecto

A primeira medida a tomar quando se começa um novo projecto é a escolha de um gestor de projecto [68]. Mas será que faz sentido escolher um gestor para pequenos projectos? Existem duas abordagens distintas para esta questão. A primeira defende que sim, que deve existir um gestor para projectos de pequena escala [20] [25] [26]. Neste caso o gestor tem a seu cargo mais do que um pequeno projecto e consequentemente não poderá dedicar-se a 100% a cada um dos projectos onde está envolvido.

A segunda abordagem defende a existência de um *juggler* em substituição do gestor de projecto [25] [26]. O *juggler* é simultaneamente gestor de projecto e elemento de desenvolvimento tendo portanto que saber gerir a equipa e realizar as suas tarefas que serão, obviamente, de menor esforço, estar consciente em cada momento do estado do sistema e da

sua evolução, e dedicar-se inteiramente aquele projecto. Para além disso, como o *juggler* possui melhores canais de comunicação para com os colegas, pode aproveitar esse facto para inculcar uma maior motivação nos mesmos.

Ambas as vertentes têm vantagens e desvantagens consoante os projectos em questão. Por exemplo, se se tratar de um pequeno projecto de 3 pessoas durante uma semana, o mais sensato será adoptar a segunda abordagem. Senão repare-se: numa equipa de 3 elementos onde um elemento é apenas gestor, apenas ficam 2/3 da equipa para efectivamente desenvolverem um projecto sendo 1/3 direccionado para a gestão de projecto. Para além disso, num projecto de uma semana será mesmo necessário 1/3 de esforço para a gestão do mesmo? Provavelmente a escolha de um *juggler* para esta situação concreta fará mais sentido pois toda a equipa aplicará esforços para o desenvolvimento do projecto. Já num caso em que o projecto envolveria 6 elementos durante 2 meses, o esforço que o *juggler* teria poderia ser diluído pelos restantes elementos da equipa e ser direccionado efectivamente para o trabalho de gestor de projecto.

3.2.2. Características

Outras características dos pequenos projectos que os distinguem verdadeiramente dos projectos de grande escala são [20] [25] [26]:

- Orientados / sensíveis ao tempo,
- Oportunistas,
- Objectivo final bem definido,
- Nem todos os requisitos estão bem definidos no início,
- Podem ser extensões a produtos já existentes,
- Riscos e custos menos elevados do que nos grandes projectos,
- Controlo e planeamento flexíveis,
- Recursos e equipamentos partilhados por mais projectos,
- Fase de desenho bastante característica,
- Redes de comunicação fortes.

Os pequenos projectos são orientados e sensíveis ao tempo dado que o período em que estes ocorrem é curto e, como tal, qualquer atraso se reflecte com maior importância nas fases críticas do mesmo; ao contrário dos grandes projectos, onde a possibilidade de reajustar a calendarização é maior. Contudo, não é boa prática encurtar o tempo destinado a outras tarefas para compensar uma tarefa atrasada pois se estas se encontrarem no caminho crítico (fluxo de tarefas que definem o traço principal do projecto e definem a data de fim do projecto), o deslize vai ter impacto em todo o projecto.

Também devido ao facto deste tipo de projectos se realizar em curtos períodos de tempo, os pequenos projectos dizem-se oportunistas pois o valor que estes criam apenas se reflecte com maior intensidade em instantes bem definidos para uma organização. O que hoje pode gerar 1.000.000, amanhã poderá apenas gerar 1.000.

A lista de requisitos vai sendo composta no decorrer de um pequeno projecto, o que não significa que os requisitos sejam conduzidos pela implementação do projecto pois seria um erro.

À medida que existam encontros com o cliente (não necessariamente externo) deve-se aperfeiçoar e detalhar a lista de requisitos mesmo que esta já esteja, à partida, concluída e, através de questões direccionadas e previamente seleccionadas, perceber se existem requisitos em falta. Este processo deve ser iterativo e sempre que se modifique algum requisito é peremptório que todas as tarefas sejam revistas quer ao nível da calendarização, quer ao nível do esforço necessário.

Analogamente à escala do projecto, os custos de um pequeno projecto são igualmente menores. É fácil perceber que, se um pequeno projecto é composto por uma equipa de, no máximo, seis pessoas com calendarizações até dois meses, então os custos associados serão inferiores ao dos grandes projectos (que são compostos, por exemplo, por uma equipa de doze elementos durante seis meses).

Em relação aos riscos de um pequeno projecto pode dizer-se que também estes têm menor impacto na maioria das vezes. Contudo, cada projecto é um caso único e aos olhos da organização um pequeno projecto pode vir a tornar-se bastante mais importante do que um projecto de grande escala. Perante esta situação a dimensão dos riscos associados ao projecto ganham uma outra importância. O que não se pode garantir é que, qualquer que seja o projecto, estes não tenha riscos associados.

Uma outra característica que deve ser tida em conta é a flexibilidade de planeamento e controlo num pequeno projecto pois é nestas fases da gestão de projecto que o gestor tem mais problemas para resolver. Alguns destes problemas devem-se a:

- O Cliente muda constantemente de requisitos,
- Os elementos da equipa têm dificuldades em estimar o trabalho que têm para fazer,
- Atrasos no projecto,
- Calendarização irrealista,
- Entrada e saída de elementos para a equipa de projecto,
- Pouco tempo, por parte do gestor, para gerir o projecto.

Sempre que ocorre alguma destas situações existem medidas a ser tomadas que implicam reajustamentos nas fases de planeamento e controlo. Deste modo, o gestor de projecto tem que estar preparado para tal e assegurar que as ferramentas de suporte de que dispõe são os ideais para o auxiliar nas acções a tomar.

3.2.3. Fase de Desenho

A fase de desenho é também bastante diferente nos pequenos projectos quando comparada com a mesma fase nos grandes projectos. Existem duas questões fundamentais para perceber as diferenças entre estas fases nos grandes e pequenos projectos [20]:

1. Deve fazer-se o desenho geral no início?
2. Qual o esforço a investir nesta tarefa?

A resposta à pergunta 1 é sim, deve fazer-se no início o desenho geral do projecto com toda a documentação necessária.

A resposta à pergunta 2 é não investir demasiado nesta tarefa pois, como já foi referido, os pequenos projectos são bastante sensíveis à mudança de requisitos. A alternativa é aperfeiçoar o desenho à medida que o projecto evolui e as certezas em relação aos requisitos vão sendo cimentadas e, só próximo do fim do projecto, criar documentação de manutenção.

Em qualquer projecto, independentemente da escala, é necessário haver fortes redes de comunicação entre os elementos que compõem a equipa de projecto. Quando os projectos em causa são pequenos, a prioridade é bastante maior dado o curto intervalo de tempo em que estes se realizam. A par que num projecto de grande escala as redes de comunicação vão sendo criadas lentamente e com um grau de ligação fraco no início, em projectos de pequena escala este cenário é impensável dado que o tempo disponível para essas redes serem criadas é nulo.

Este é dos pontos a ter mais em conta pelo gestor de projecto aquando da escolha dos restantes elementos da equipa, pois são as pessoas e a relação entre elas que contribuem em grande parte para os bons resultados dos projectos. Assim, construir uma equipa onde as redes de comunicação já existem e são consistentes vai claramente facilitar todo o processo de gestão de problemas.

O gestor de projecto deverá também procurar por inteligência, motivação e experiência nos elementos da equipa. Qualquer uma destas características é importante em cada elemento pois basta faltar motivação, para que uma pessoa já não apresentar o rendimento esperado.

Ainda no seguimento das redes de comunicação faz sentido esclarecer também que num pequeno projecto o factor participação pode influenciar bastante o desenvolvimento do

projecto. Nas fases em que o gestor não é capaz de dar resposta a todos os problemas em tempo útil, é necessário tomar medidas que permitam ao projecto continuar a sua evolução.

Uma dessas medidas passa por quebrar a hierarquia existente entre cliente, gestor e restantes elementos da equipa para criar canais de comunicação entre o cliente e toda ou parte da equipa. Os elementos de desenvolvimento passam a ter um papel mais activo e com responsabilidades acrescidas. Como tal, cabe ao gestor perceber se os elementos da equipa são capazes de desempenhar papéis como este quando efectua as entrevistas de recrutamento para, em fases críticas, fazer as escolhas certas.

Associado a todos estes factores surge a partilha de informação. Num pequeno projecto é importante que seja estudado e criado um método que permita a troca fácil e consistente de informação entre os vários elementos da equipa. É incomportável haver reuniões sempre que se queira fazer um ponto de situação do projecto. A resposta rápida é crucial quando estamos perante situações onde o tempo é curto e, se for necessário esperar pela próxima reunião para saber qual a próxima tarefa a realizar, o mais certo é que o projecto se atrase. Não quero com isto dizer que reuniões semanais entre a equipa e o cliente não sejam necessárias, pelo contrário. Manter o cliente e a própria equipa informados é também um requisito essencial para o bom funcionamento e evolução do projecto.

3.3. Metodologias e Ferramentas

Posteriormente aos problemas apresentados sobre pequenos projectos em si e as suas características, existem outros que os afectam directamente e que dizem respeito às metodologias e às ferramentas de software de suporte à gestão de projectos utilizadas para gerir todo este processo.

3.3.1. Metodologias

Foram descritas no subcapítulo 2.3.1 algumas metodologias que, ao longo dos anos, têm sido desenvolvidas no sentido de ajudar os gestores de projecto a melhor conduzirem todo o processo da gestão de projectos. Em todas as metodologias ou boas práticas existem condicionantes impeditivas de as transpor para o lado dos pequenos projectos, mesmo dentro da área de projectos de TI/SI.

Muitos gestores tentaram adoptar as técnicas usadas em grandes projectos na gestão de múltiplos pequenos projectos [20] [7] [39] [40], sendo que os resultados foram comprometedores. Em primeiro lugar, a equipa que compõe um grande projecto é, ou deveria ser, composta por alguns especialistas em, por exemplo, planeamento ou estimativa de custos. Nos pequenos projectos não existe esta possibilidade devido ao reduzido número de elementos.

Os métodos genéricos de gestão de projectos chegam a níveis de detalhe de planeamento e estimativa de custos a que não é possível chegar nos pequenos projectos, como já foi referido, devido ao curto ciclo de vida dos mesmos. Como tal, não existe uma base suficientemente forte e detalhada para que possam ser aplicados os métodos de controlo comuns aos pequenos projectos.

Outro factor preponderante para a não aplicação das metodologias comuns a pequenos projectos é que estas técnicas não estão preparadas para lidar com partilha de responsabilidades, curtas calendarizações ou para o problema da múltipla gestão de projectos.

Contudo, o principal motivo pelo qual as metodologias comuns não funcionam nos pequenos projectos deve-se ao facto dos pequenos projectos serem, em muitos casos, extensões/modificações a antigos projectos (também conhecidos por projectos de manutenção) e, como tal, todos os processos já mencionados (planeamento, estimativa de custos, gestão de recursos) são ainda mais difíceis de definir.

Como resultado da aplicação dos métodos generalistas de gestão de projecto a pequenos projectos surgem muitas das vezes problemas mais específicos [7]:

- Sobrecarga na documentação,
- Falta de tempo para revisões,
- Elevada requisição de recursos,
- Custos de treino elevados, devido à relação tempo disponível e complexidade associada às metodologias,
- Falta de orientação na equipa.

Posto isto, pode concluir-se que pequenos projectos necessitam de técnicas específicas de gestão e ferramentas que dêem suporte a todo este processo. Este é, sem dúvida, o maior desafio que se impõe, dado que ao longo dos anos continuam sem resposta à altura.

3.3.2. Ferramentas

As ferramentas de software existentes no mercado para facilitar a tarefa de gestão de projectos são, na sua maioria, eficazes na gestão de grandes projectos. No entanto, como foi apresentado na Tabela 3, existem uma série de requisitos onde estas ferramentas falham. Estes requisitos são um pouco a imagem dos problemas dos pequenos projectos que foram identificados no início deste capítulo. Ora vejamos:

- Facilidade de utilização e treino: A maioria das ferramentas existentes é complicada de utilizar o que se torna um entrave à variável Tempo, que neste tipo de projectos é fundamental.

- Funcionalidades de desenho e planeamento do projecto directas: Este é o requisito menos crítico nas ferramentas existentes, apesar de ser essencial para o gestor de projecto. Existem no entanto ferramentas que não apresentam esta característica.
- Suporte à comunicação: Como foi referido no subcapítulo anterior, é dos pontos mais críticos num pequeno projecto e é, de modo geral, o requisito mais fraco ou mesmo inexistente na maioria das aplicações de gestão de projectos. É necessário que qualquer membro da equipa tenha acessibilidade a um plano de projecto centralizado e actualizado e que, sempre que haja alterações este seja notificado sem necessidade de que o gestor de projecto redistribua o plano membro por membro. Mais uma vez, o factor Tempo é aqui posto em causa.
- Gestão de recursos partilhados: Este é outro dos factores críticos nas ferramentas de gestão de pequenos projectos. É muito natural viver-se um ambiente multi-projecto nas empresas onde os recursos são obrigatoriamente partilhados. Como haverá um gestor de projecto, que está a formar a equipa de projecto, saber quais as pessoas que estão neste momento livres? A extensão EPM (*Enterprise Project Management*) do *Microsoft Project* é, mais uma vez, a solução mais utilizada. No entanto, a taxa de adesão dos utilizadores desta ferramenta é baixa devido ao elevado período de tempo que é necessário apenas para reportar o final de uma actividade. Incomportável para um pequeno projecto.
- Manter um banco de dados e informação acessível a todos os elementos do projecto: É imperativo que o fluxo de informação seja em tempo real e acessível a cada elemento da equipa no momento em que este necessite da mesma. Como tal, as ferramentas devem suportar este requisito, coisa que, mais uma vez, não acontece. Existem como alternativa ferramentas de gestão documental mas que têm a desvantagem de ser mais um sistema a utilizar, mais uma competência a desenvolver e mais tempo a perder na aprendizagem de outra ferramenta.

Em suma, a gestão de pequenos projectos está longe de ser uma disciplina de sucesso dado que não existe uma metodologia nem uma ferramenta de suporte à gestão que responda exactamente às necessidades dos pequenos projectos. O que existe são aproximações ou abordagens incompletas que têm vindo a comprometer a área dos SI / TI, bem como a acreditação da mesma nas empresas.

3.4. Problemas da Gestão de Pequenos Projectos

São inúmeros os problemas relacionados com a gestão de pequenos projectos. Um dos maiores problemas nos pequenos projectos acaba por ser a pouca importância que lhes é atribuída por parte das organizações, daí muitas vezes estes resultarem em fracassos. O facto dos custos associados aos pequenos projectos serem também pequenos (por projecto) faz com que a sua importância seja também subestimada.

Contudo, ao contrário do que se passa nos projectos de grande escala, o maior problema associado a pequenos projectos é a gestão de múltiplos pequenos projectos.

Tabela 4. Problemas básicos da gestão de múltiplos pequenos projectos (Fonte: [20])

	Pequenos Projectos	Grandes Projectos
Nº. de Projectos	20	1
Nº. de Calendarizações e Estimativas	20	1
Duração Média	1 Mês	1 Ano e 6 Meses
Gestão a Tempo Inteiro	Não	Sim

Os dados apresentados na Tabela 4 traduzem a ideia de muitos projectos, em pouco tempo, e com atenção partilhada. O gestor de projecto não tem como se dedicar exclusivamente a um projecto pois tem a seu cargo 20 onde, por vezes, tem que partilhar os recursos. Também por este motivo a dificuldade das estimativas aumenta pois não se pode pensar num projecto isoladamente.

Segundo alguns autores gerir um pequeno projecto é a combinação de três grandes conjuntos de problemas: satisfação do cliente, comunicação e problemas técnicos e de gestão [69] [36].

O primeiro problema está relacionado com o facto da alteração de requisitos e avaliação dos mesmos junto do cliente. Como já foi referido, em projectos de pequena dimensão, qualquer modificação aos requisitos tem impactos consideráveis na gestão e planeamento do projecto. Porém, sempre que haja, por parte do cliente, necessidade de os modificar, a equipa de projecto terá que analisar as repercussões que essas alterações irão trazer e consciencializar o cliente do mesmo. Apenas quando ambas as partes concordarem com as modificações ao projecto estas devem ser levadas para a frente. Daí o facto de, em pequenos projectos, inicialmente não se dever fazer especificações rígidas, mas antes viradas para a mudança.

O segundo problema lida com a comunicação entre os elementos da equipa. O ambiente de desenvolvimento num pequeno projecto obriga a que existam fortes canais de comunicação entre os elementos, senão vejamos:

- Sempre que existam dificuldades por parte dos elementos da equipa no desenvolvimento do projecto é incomportável que o gestor de projecto atenda a todos e todos os dias. Não há tempo para tal, mas o projecto não pode parar e os elementos da equipa não podem esperar pela próxima reunião para colocarem as suas questões. É necessário que os elementos da equipa comuniquem entre si e partilhem experiências e situações que ajudem os outros elementos a resolver o mesmo tipo de problemas.
- Sobrecarga de gestão: em pequenos projectos esta é uma situação bastante provável devido ao reduzido número de elementos. O gestor de projecto antecipa-se a resolver problemas que ainda não surgiram em vez de concentrar os seus esforços noutras tarefas e deixar que os próprios elementos da equipa o solicitem quando necessário. Uma boa rede de comunicação tem aqui um contributo essencial.
- Sub-gestão é o problema contrário e surge quando o gestor exerce demasiado trabalho de desenvolvimento (caso em que assume o papel de *juggler*) ou de gestão (está responsável por demasiados projectos) e fica com pouco tempo para dedicar à equipa e comunicar com ela. Nestas situações deve ponderar-se o trabalho atribuído ao gestor e garantir que, dentro de cada projecto, ele dispõe de 45% do tempo para dedicar ao processo de gestão.
- Introdução de novos elementos na equipa cria problemas na comunicação dado que n-1 redes de comunicação terão que ser criadas. Quando um projecto já está em fases avançadas, este processo torna-se penoso pois nem os elementos de equipa podem largar as suas tarefas, nem o novo elemento pretende dispensar tempo para essa tarefa em vez de se dedicar à integração no desenvolvimento.

O terceiro problema é um pouco mais complexo e abrangente. Lida com problemas de planeamento, técnicos e de controlo.

Os problemas técnicos devem inicialmente ser tratados pelas vias referidas no ponto anterior. Só em casos extremos o gestor deve dedicar-se à resolução do problema junto do elemento da equipa em causa.

Os problemas de planeamento e controlo [36] são um pouco mais complexos e a sua resolução nem sempre segue um conjunto de actividades bem definido. Este tipo de problemas depende muito de factores externos e dos próprios membros da equipa de projecto.

O planeamento é sempre uma actividade ingrata e muito difícil para quem a executa. Esta actividade é, na maioria dos casos, baseada na experiência dos gestores de projectos perante o tipo de actividades que têm de planear. É natural que, sendo uma pessoa a planear as tarefas de uma outra, a estimativa não venha muito correcta.

Existem alguns métodos de que ajudam a planear tarefas. A análise PERT (*Program Evaluation and Review Technique*) ou CPM (*Critical Path Method*) são ferramentas bastante úteis para determinar o tempo necessário para concretizar cada tarefa e, no final, identificar o período mínimo necessário para terminar o projecto. Contudo, os *inputs* destas ferramentas são, mais uma vez, fortemente condicionados pelo gestor de projecto e não pelo responsável pela execução de uma dada tarefa.

Ao nível dos pequenos projectos de TI o panorama é um pouco mais complexo e o impacto maior devido a:

- Períodos curtos de execução das tarefas,
- Maior número de tarefas em paralelo distribuídas por vários recursos que partilham vários projectos,
- Acrescida complexidade em determinar a duração de actividades de desenvolvimento de software.

Ao nível do controlo, os aspectos mais evidentes são:

- Curto ciclo de vida do projecto: Este factor leva a que reste pouco tempo para que se possa recolher informação acerca do projecto, identificar problemas e corrigi-los.
- Responsabilidades partilhadas: Torna-se difícil garantir que todos os elementos envolvidos cumpram com os seus compromissos e que tenham prontas a tempo as entregas marcadas na calendarização.
- Problemas em obter informação actual: Este problema resulta do processo inadequado de reportar informação. Na maioria das vezes os elementos da equipa não têm acesso à informação ou quando têm já é tarde para que esta valha alguma coisa. Por este motivo é necessário que se defina antes do arranque do projecto um mecanismo que permita centralizar os dados e garantir que estes sejam fornecidos a tempo a todos os elementos da equipa.
- Controlo de múltiplos projectos: Um pequeno projecto por si só acaba por não ter um controlo muito complexo. Mas se a este projecto juntarmos mais dez, seguramente que a complexidade de controlo vai aumentar pelo menos na mesma proporção. Para além do número de projectos em simultâneo surge ainda o facto de cada projecto se poder encontrar numa fase diferente de todos os outros. Este factor acaba por se tornar numa barreira ao desempenho do gestor pois não haverá a possibilidade de reaproveitar trabalho de um projecto para outro. Outro ponto crítico da gestão de múltiplos pequenos projectos é a gestão de recursos pois, dentro de uma organização, os recursos são partilhados pelos vários projectos que decorrem num determinado período. Os problemas associados à gestão de múltiplos projectos podem dividir-se em cinco categorias [70]:

- Complexidade: na gestão de recursos em ambientes de múltiplos projectos,
- Capacidade: habilidade que a organização tem de suportar múltiplos projectos e garantir suficientes e apropriados recursos,
- Conflitos: das pessoas, dos sistemas e da organização,
- Compromissos: relacionados com os compromissos dos elementos de cada projecto individualmente e das entidades responsáveis em providenciar os recursos,
- Contexto: relacionado com as configurações do projecto, como procedimentos, cultura ou normas de comportamento.

3.5. Conclusão

O paradigma da empresa orientada a projectos é cada vez mais comum no sector das TI. Os projectos tendem a ser cada vez mais curtos pois, o factor tempo, é para muitas organizações vantagem competitiva face às suas concorrentes. Porém, não existem ainda metodologias ou ferramentas (software) desenhadas propositadamente para este tipo de projectos e que contemplem os requisitos necessários para que estes se tornem realmente úteis aos gestores de projectos. Há, portanto, uma falha grave nesta disciplina que merece uma atenção especial devido aos largos milhares que nela são perdidos todos os anos.

4. Proposta

Neste capítulo pretende-se formular uma proposta que dê resposta ao problema apresentado. Assim como a definição do problema, também a proposta vai ser a apresentada segundo as duas áreas de análise: Metodologias e Ferramentas da gestão de pequenos projectos.

4.1. Metodologia

Um dos principais motivos pelos quais os pequenos projectos de SI / TI são um fracasso é a inexistência de uma metodologia desenhada exclusivamente para pequenos projectos.

Todas as abordagens de aproximação de metodologias desenhadas para grandes projectos apresentam uma série de problemas já identificados e detalhados no capítulo 3 deste documento. A partir desse conjunto de problemas e das características próprias dos pequenos projectos, foi possível identificar uma série de requisitos para a correcta definição de uma metodologia de suporte à gestão de pequenos projectos [20] [71]:

- **Simples:** A tarefa de gerir um projecto já é bastante complicada, logo as metodologias envolvidas devem ser fáceis de perceber por si e de pôr em prática, sem necessidade de treino.
- **Alinhada com o período de vida típico de pequenos projectos.**
- **Uma abordagem comum:** Os pequenos projectos, como os grandes projectos, variam bastante. O objectivo deste critério é garantir que qualquer técnica possa ser adoptada por qualquer tipo de projecto e sem necessidade de redefinição de processos.
- **Contemplar ambientes multi-projecto de modo flexível** ao ponto de encaixar com os processos utilizados na organização em causa.
- **Não pode acrescentar complexidade alguma** quer aos processos, quer às interfaces organizacionais, pelo contrário, deve facilitar e/ou adaptar-se aos mesmos e não ao contrário.
- **Deve contemplar todos os processos e ferramentas necessárias** para as tarefas mais comuns da gestão de projectos:
 - **Planeamento** – atendendo exactamente às calendarizações típicas de um pequeno projecto e à sua revisão. Como tal, este processo deve ser constantemente posto em prática, no limite diariamente.
 - **Comunicação** – quer ao nível dos canais de comunicação, quer ao nível das entidades informacionais que circulam entre os *stakeholders* do

projecto, este processo deve ser o mais eficiente e simples de operacionalizar.

- Risco – manter todas as actividades deste processo pois o facto de um pequeno projecto ser em dimensão mais reduzido, isso não implica que os riscos e a probabilidade de ocorrerem sejam menores nem o impacto destes no projecto. No entanto este processo deve ser agilizado de modo a ele próprio não atrasar o projecto propriamente dito.
 - Controlo – deve ser um procedimento constante e mensurável de modo a ser feita a avaliação do projecto a cada etapa.
 - Encerramento – deve apenas ser uma formalização do projecto, pois todo o trabalho de avaliação do projecto deve ser executado a cada tarefa por cada um dos intervenientes.
 - Resposta rápida: A pequena duração deste tipo de projectos e a rapidez com que os requisitos mudam, requerem métodos que providenciem informação e respostas rápidas e correctas em qualquer momento do projecto.
- Promover um ambiente colaborativo entre toda a equipa de projecto de modo a que todos os elementos se sintam parte activa no mesmo.
 - Deve acompanhar-se todo o processo de gestão de projecto com o auxílio de computadores e aplicações que o simplifiquem.

Partindo dos requisitos listados e com suporte na bibliografia, esta proposta estende-se a uma metodologia concreta. O processo de trabalhos assente na mesma está representado na Figura 15 onde estão marcadas com cores diferentes quer as actividades (cor rosa), quer os pontos de decisão (cor verde) que deverão ser tomados em colaboração.

Existem 3 tipos de actores que actuam directamente na gestão de pequenos projectos: o cliente, o maior interessado no sucesso do projecto e aquele que decide acima do gestor de projecto, o gestor de projecto que desempenha a função de “maestro” do projecto e coordena a equipa de projecto que, analogamente à metáfora anterior, podem ser vistos como os “violinistas”.

O processo que suporta a metodologia começa pela acção do cliente. Este propõe um novo projecto especificando o seu âmbito. O gestor de projecto faz a primeira revisão e, caso alguma especificação esteja menos clara, o gestor de projecto parte para um processo iterativo de esclarecimento com o cliente para garantir que, quando a especificação chegar à equipa, não haja dúvidas.

Na actividade seguinte, o gestor de projecto em colaboração com os restantes elementos da equipa estimam tempo, custos e esforço. Esta actividade é de carácter colaborativo dado que é nestas actividades que normalmente começam os problemas dos pequenos projectos. Tipicamente, é o gestor de projecto o responsável pela execução desta tarefa, mas falta de

experiência em estimativas ou falta de conhecimento das capacidades dos elementos da equipa de projecto, levam a que este cometa erros com enorme impacto no projecto. A decisão de continuar com o projecto ou não, é igualmente tomada em conjunto e comunicada ao cliente, de forma a que este reveja as especificações do projecto e a sua prioridade, caso a decisão seja de não avançar com o projecto. Após a revisão, o cliente em colaboração com o gestor de projecto, valida a possibilidade de avançar com o projecto e, caso ainda seja impossível, então o projecto é colocado numa “lista de espera” consoante a sua prioridade e disponibilidade.

Caso o projecto seja viável, então a equipa de projecto efectua uma revisão final das especificações e começa a execução do projecto. Esta é mais uma actividade colaborativa onde a interacção entre gestor de projecto e equipa pode ser tanto maior quanto o número de tarefas de execução o gestor apresentar. Se a abordagem escolhida para a gestão de um projecto for a de um *juggler* (abordagem apresentada no capítulo 3.2.1), então a proximidade destas duas entidades é maior e, conseqüentemente, a colaboração entre os mesmos. Se a abordagem seguida corresponder à de um gestor com funções em multi-projectos, certamente que haverá menor colaboração.

Porém, o gestor de projecto terá sempre que, paralelamente, controlar o projecto e propor, sempre que necessário, alterações ao planeamento para validar com a equipa. Como foi referido no capítulo 3.2.1, o gestor de projecto, independentemente da abordagem seguida, deverá reservar, consoante as necessidades, um período para assegurar o cumprimento destas tarefas.

Por fim, após a execução do projecto, deverá ser efectuado o fecho do projecto. Nesta actividade participam todos os envolvidos no processo, incluindo o cliente. É nesta fase que são levantados os pontos positivos e negativos do projecto e onde se registam os mesmos de modo a serem contemplados no projecto seguinte.

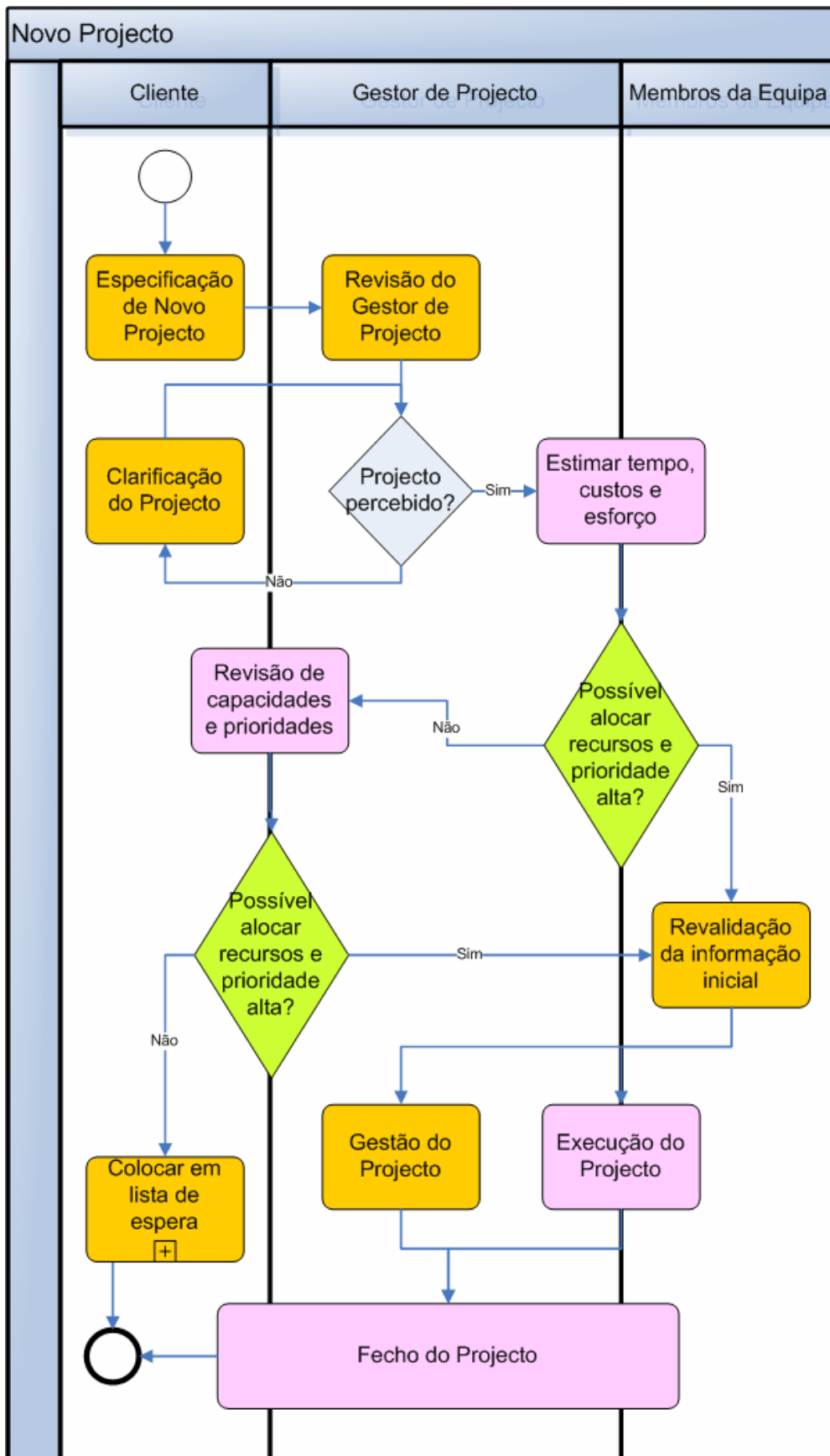


Figura 15. Metodologia proposta

4.2. Requisitos da Ferramenta

Se por um lado não existe uma metodologia focada em pequenos projectos, por outro, as ferramentas de software existentes também não vão ao encontro das necessidades dos pequenos projectos, não de forma única e centralizada.

A proposta para a definição de uma ferramenta de gestão de pequenos projectos surge através do cruzamento entre as necessidades de gestão dos mesmos com os requisitos funcionais das ferramentas existentes. Dessa análise resultou uma nova lista de requisitos funcionais que, por um lado vai de encontro com alguns dos requisitos já implementados pelas ferramentas existentes e outros resultantes da minha pesquisa [20] e, por outro lado, inclui uma série de outras características resultantes das falhas das referidas ferramentas [21].

Logicamente, todos os requisitos desta ferramenta estão de acordo com a definição da metodologia do subcapítulo 4.1. Todavia, o mapeamento entre estas duas componentes não é directo até porque existem processos que não são implementados pela ferramenta em si, mas antes pela própria equipa de projecto.

O levantamento de requisitos foi um processo incremental que foi sendo composto à medida que foram sendo observadas e avaliadas as várias ferramentas.

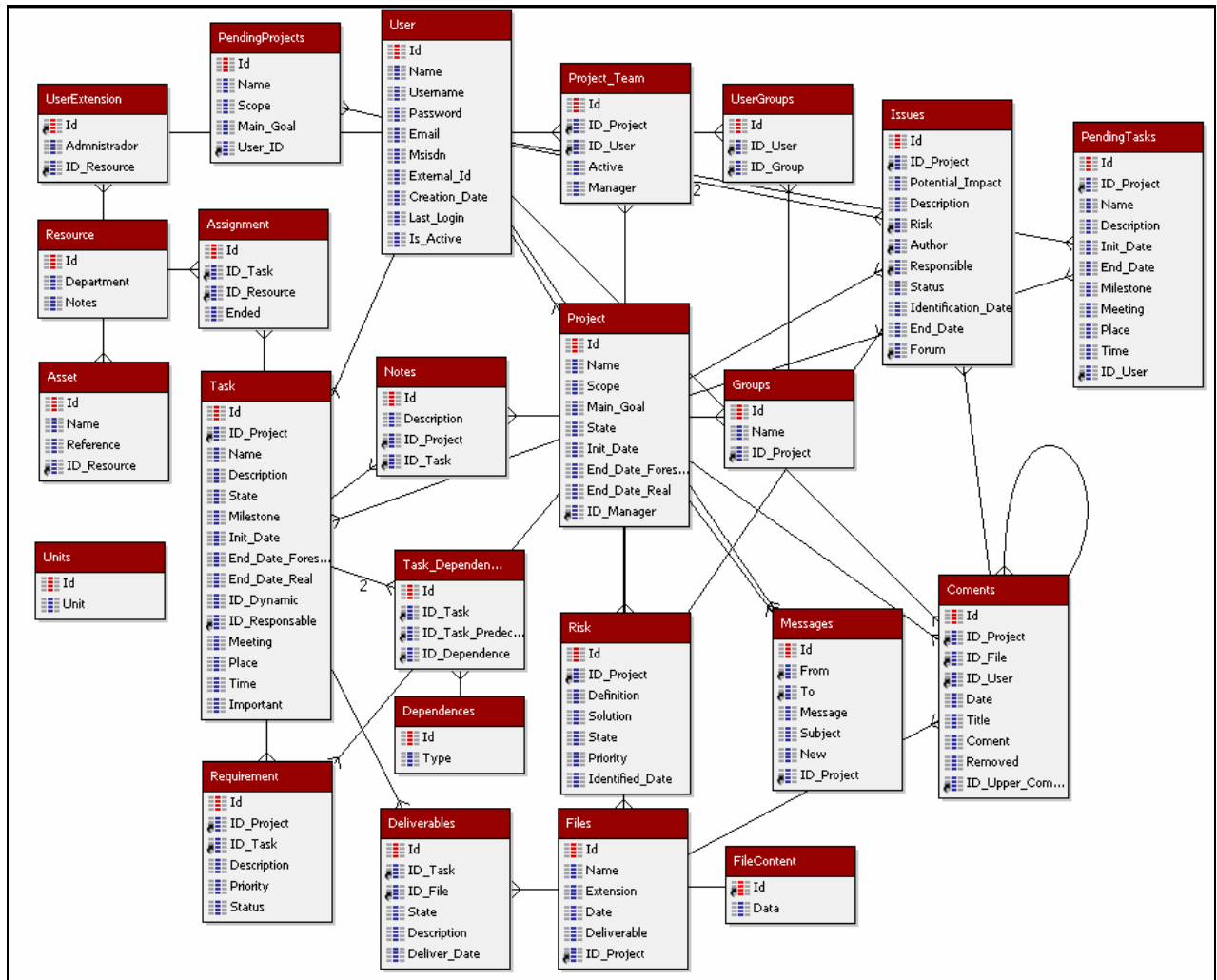
Para pequenos projectos existem características/funcionalidades mais específicas que devem ser contempladas por estas ferramentas tendo por base algumas das funcionalidades das ferramentas tradicionais usadas para projectos de grande escala. A lista de requisitos produzida, bem como o modelo de domínio encontram-se nos subcapítulos seguintes.

4.2.1. Perfis de Utilizador

Antes da listagem de requisitos funcionais da proposta, faz sentido definir os vários tipos de utilizadores da ferramenta visto que estes têm diferentes papéis, diferentes privilégios e diferentes permissões. Os diferentes tipos de utilizadores identificados são:

- Gestor de Projecto: É um utilizador comum da ferramenta. No entanto, perante um projecto de que é gestor, tem acessos e funcionalidades que os restantes elementos da equipa não têm. É o gestor de projecto que define os restantes elementos da equipa de projecto.
- Elemento da equipa de projecto (não gestor): Partilha de todas as vistas e da maioria das funcionalidades de um projecto. Contudo, há limitações no sentido de haver uma maior responsabilização das tarefas pelos vários elementos.
- Administrador do sistema: É o moderador da ferramenta e é a seu cargo que fica toda a componente de configurações de *Backoffice* e aprovações.

4.2.2. Modelo de Domínio



4.2.3. Requisitos Funcionais

Tabela 5. Requisitos Funcionais de uma ferramenta de software de gestão de pequenos projectos

RF1 – A aplicação deve incluir funcionalidades de gestão de recursos para múltiplos projectos tendo em conta que vários projectos podem partilhar o mesmo recurso.
RF2 – O número de projectos que cada gestor deve gerir em simultâneo deve também ser controlado a partir da aplicação.
RF3 – A aplicação deve incluir funcionalidades de planeamento, criar tarefas, criar

dependências entre tarefas e atribuir responsabilidades aos elementos da equipa.
RF4 – A aplicação deve mostrar a cada momento informação que permita aos elementos da equipa perceber o estado do projecto (tarefas concluídas, esforços, progresso) com suporte a gráficos de planeamento e tabelas.
RF5 – Deve ser possível assinalar marcos e entregas directamente na aplicação.
RF6 – A aplicação deve incluir funcionalidades de gestão de riscos e <i>issues</i> .
RF7 – Deve possibilitar a gestão da lista de requisitos com versões consoante as alterações feitas durante o processo de negociação com o cliente.
RF8 – Incluir meios de comunicação centralizados e que proporcionem um ambiente colaborativo (p. ex. fóruns, mensagens, etc).
RF9 – Fornecer espaços para guardar documentação a que todos os elementos da equipa de projecto tenham acesso e possam partilhar.
RF10 – Informação compactada acerca dos projectos em curso, se estão atrasados ou dentro dos limites temporais (p.ex. <i>dashboard</i>).
RF11 – Informação acerca das tarefas em curso, se estão atrasadas, dentro dos limites temporais ou prestes a terminar (p.ex. <i>dashboard</i>).
RF12 – Disponibilizar uma calendarização do género <i>to-do list</i> de modo a que cada utilizador possa ter uma perspectiva global de todos os seus projectos resultante do cruzamento da informação das várias tarefas. Permitir filtrar as tarefas de modo a criar vistas consoante as necessidades do utilizador.
RF13 – Facultar vistas temporais, por exemplo gráficos gantt.
RF14 – Notificações para os utilizadores do sistema acerca das tarefas que têm para terminar, das tarefas atrasadas, das actualizações efectuadas por outros elementos da equipa e do fim das tarefas e projectos em que participa.

4.3. Processos

À medida que a lista de requisitos se compunha, houve necessidade de dar seguimento à mesma e definir alguns fluxos de dados de modo a ditar algumas formas de utilização de uma ferramenta genérica para gestão de pequenos projectos. O melhor modo de materializar estes fluxos é através de processos descritos em notação BPMN.

Os processos mais relevantes são os de criação e fecho de projectos e tarefas que, como mostram as figuras seguintes, se podem fazer segundo algumas alternativas, dependendo da cultura das organizações.

4.3.1. Criar um Pequeno Projecto

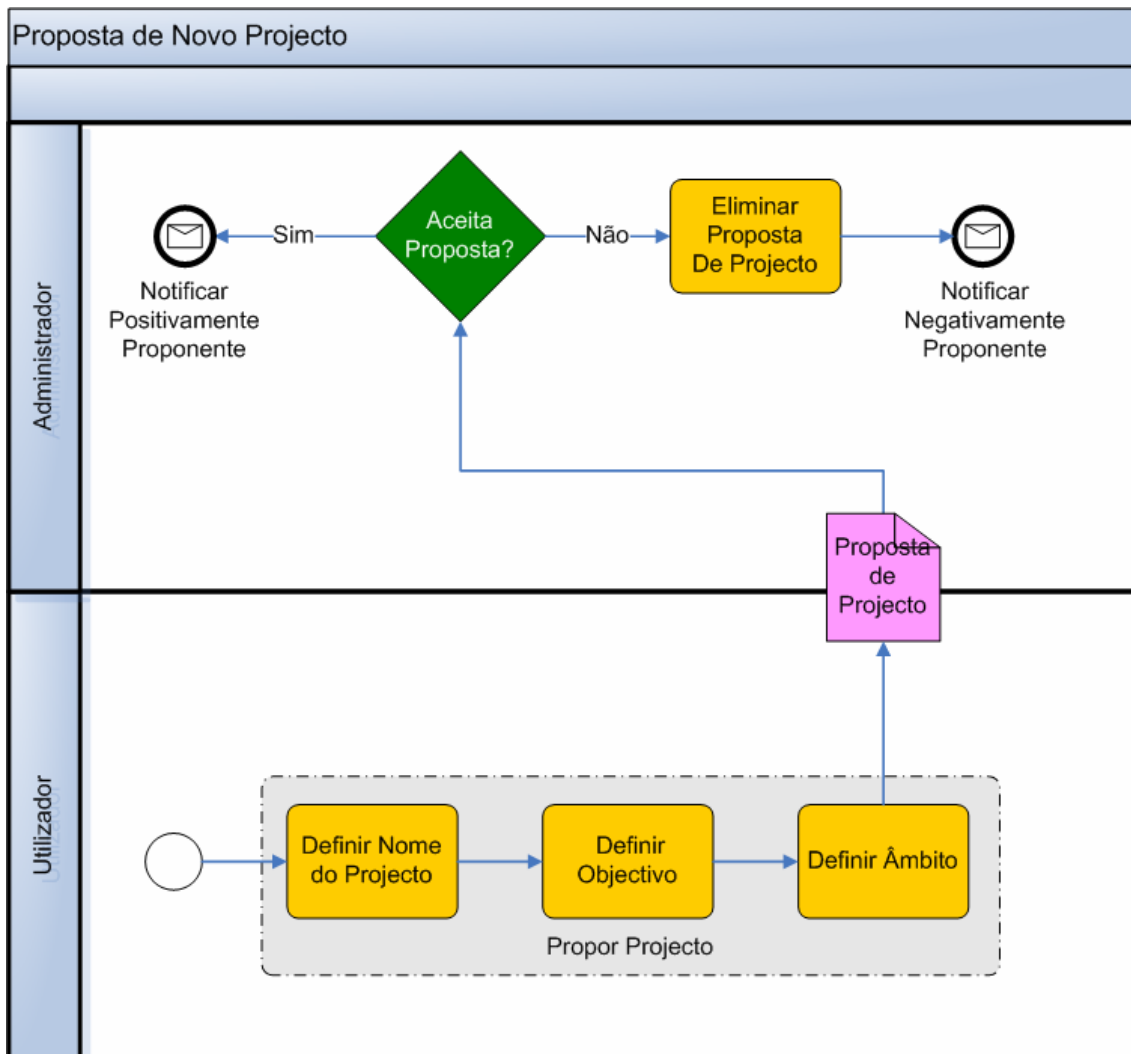


Figura 16. Criação de um Pequeno Projecto (proposta efectuada por um utilizador da aplicação)

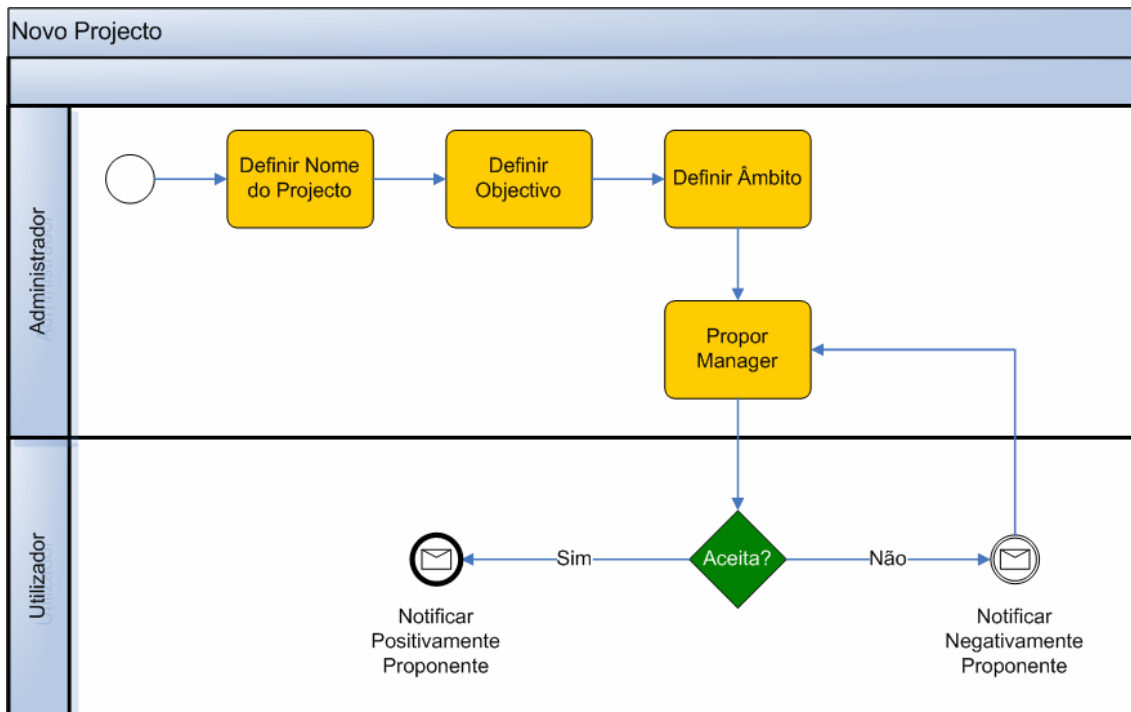


Figura 17. Criação de um Pequeno Projecto (proposta efectuada pelo administrador da aplicação)

Existem duas abordagens possíveis para a criação de um novo Pequeno Projecto. A primeira alternativa (Figura 16) é a proposta de um pequeno projecto por parte de um colaborador da empresa que, ao efectuar-la, está a assumir-se como gestor do projecto.

Após o preenchimento do nome do projecto e da sua finalidade (âmbito e objectivo) por parte do proponente, o administrador da aplicação aceita ou rejeita a proposta (esta acção deverá ser vista como moderação para evitar o uso inadequado da ferramenta). Caso a proposta seja aceite, o agora gestor do projecto está em condições de poder construir a equipa de projecto e efectuar o planeamento do projecto.

A outra alternativa para se abrir um novo projecto passa pelo administrador do sistema criar o projecto (nome do projecto e finalidade) e atribuí-lo a um gestor (Figura 17). O gestor está à partida definido, no entanto, na eventualidade de haver erros na atribuição por parte do administrador, o gestor do projecto deverá aceitar ou rejeitar a proposta perante o sistema.

4.3.2. Fecho de um Pequeno Projecto

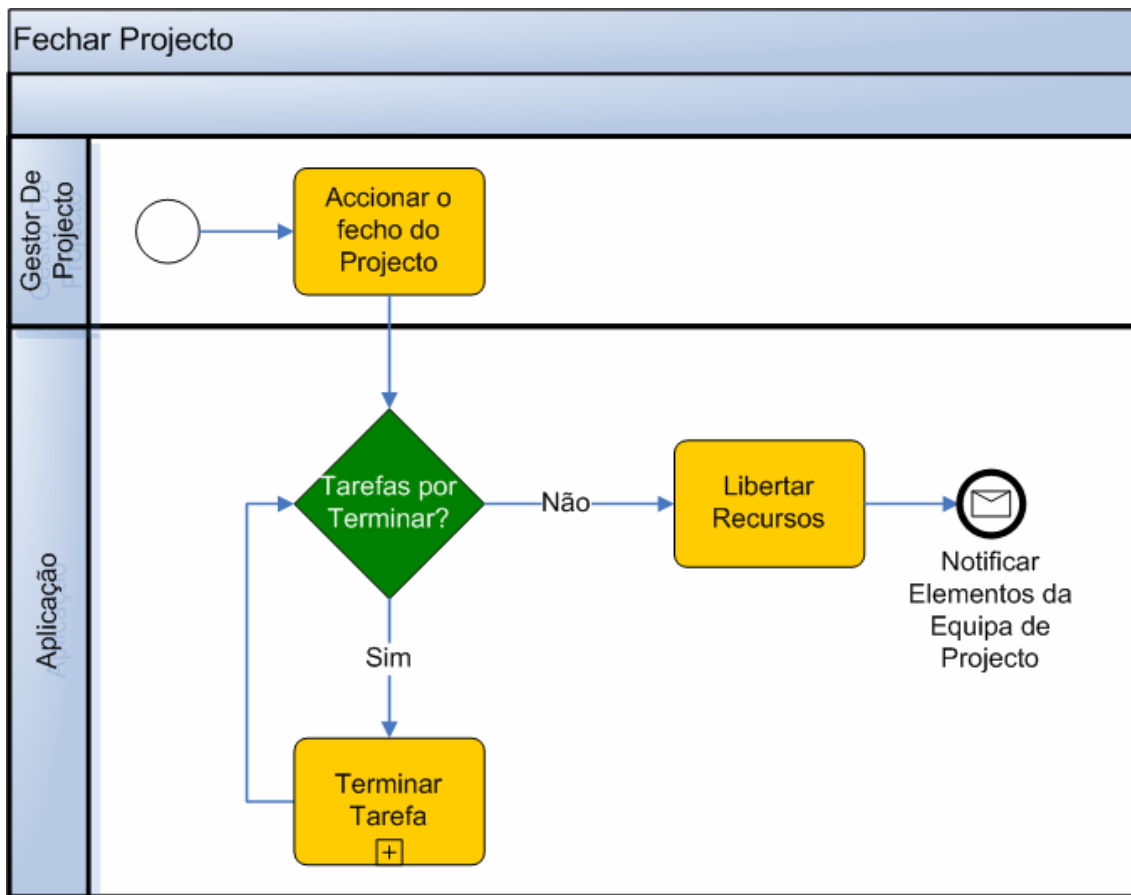


Figura 18. Fecho de um Pequeno Projecto

O fecho de um projecto é um processo bastante simples como mostra a Figura 18. Porém, é uma tarefa que não deve ser esquecida. Do ponto de vista de uma aplicação que gere recursos partilhados, é de extrema importância manter a informação actualizada a cada momento para que qualquer gestor de projectos possa saber com quem contar. Como tal, só o fecho do projecto garante que os recursos são libertados, quer para a aplicação, quer para os restantes colaboradores de uma organização.

4.3.3. Criar Nova Tarefa

Analogamente à criação de projectos, também a criação de tarefas pode seguir dois caminhos distintos: criação de tarefas por parte do gestor de projecto e proposta de um dos elementos da equipa de projecto.

Na primeira abordagem a tarefa é imediatamente criada (Figura 19). Ao criar a tarefa, o gestor de projecto tem que preencher o nome da mesma, a descrição de modo a que os restantes elementos da equipa percebam o fim da tarefa, a data de início prevista, a data de fim prevista e o responsável pela execução da tarefa. A partir do momento da submissão da tarefa, esta fica imediatamente disponível para que o seu responsável ou o gestor do projecto possam delegar trabalho.

O método alternativo surge como meio de colaboração entre a equipa (Figura 20). Qualquer elemento pode propor tarefas para o projecto. O moderador, neste caso, é o gestor de projecto que aceita ou recusa as propostas consoante o facto de esta poder estar alinhada ou não com a finalidade do projecto. Caso a proposta seja aceite, o responsável da mesma é automaticamente o proponente. Para que este tipo de colaboração entre a equipa possa ser possível, a ferramenta deverá providenciar mecanismos eficientes de comunicação.

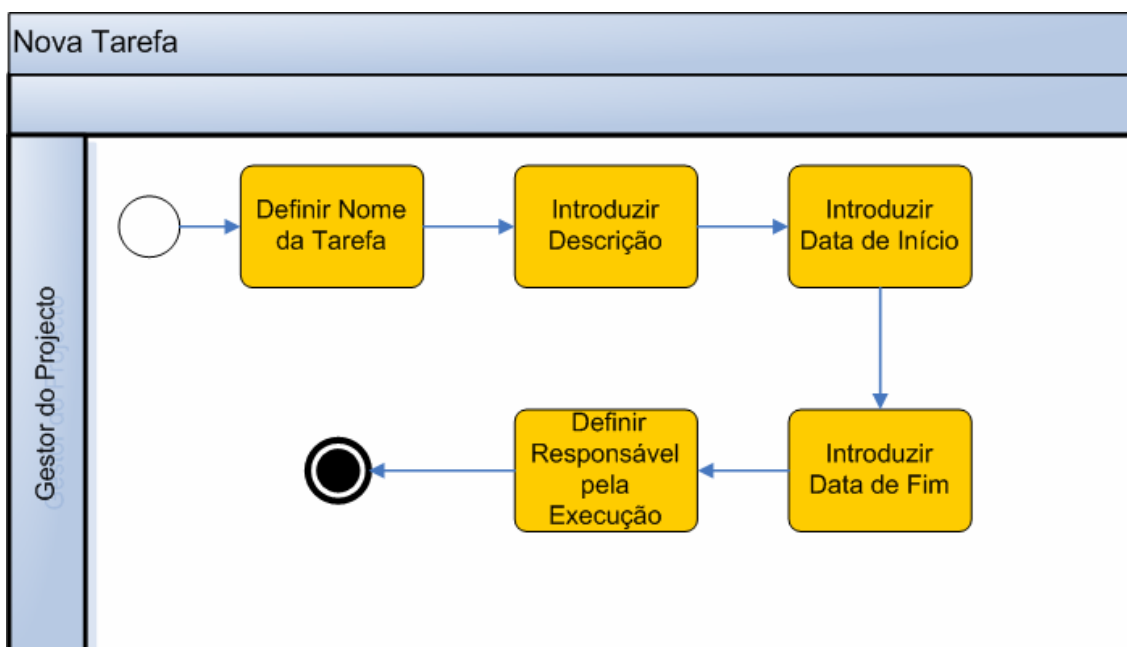


Figura 19. Criação de Nova Tarefa (Gestor de Projecto)

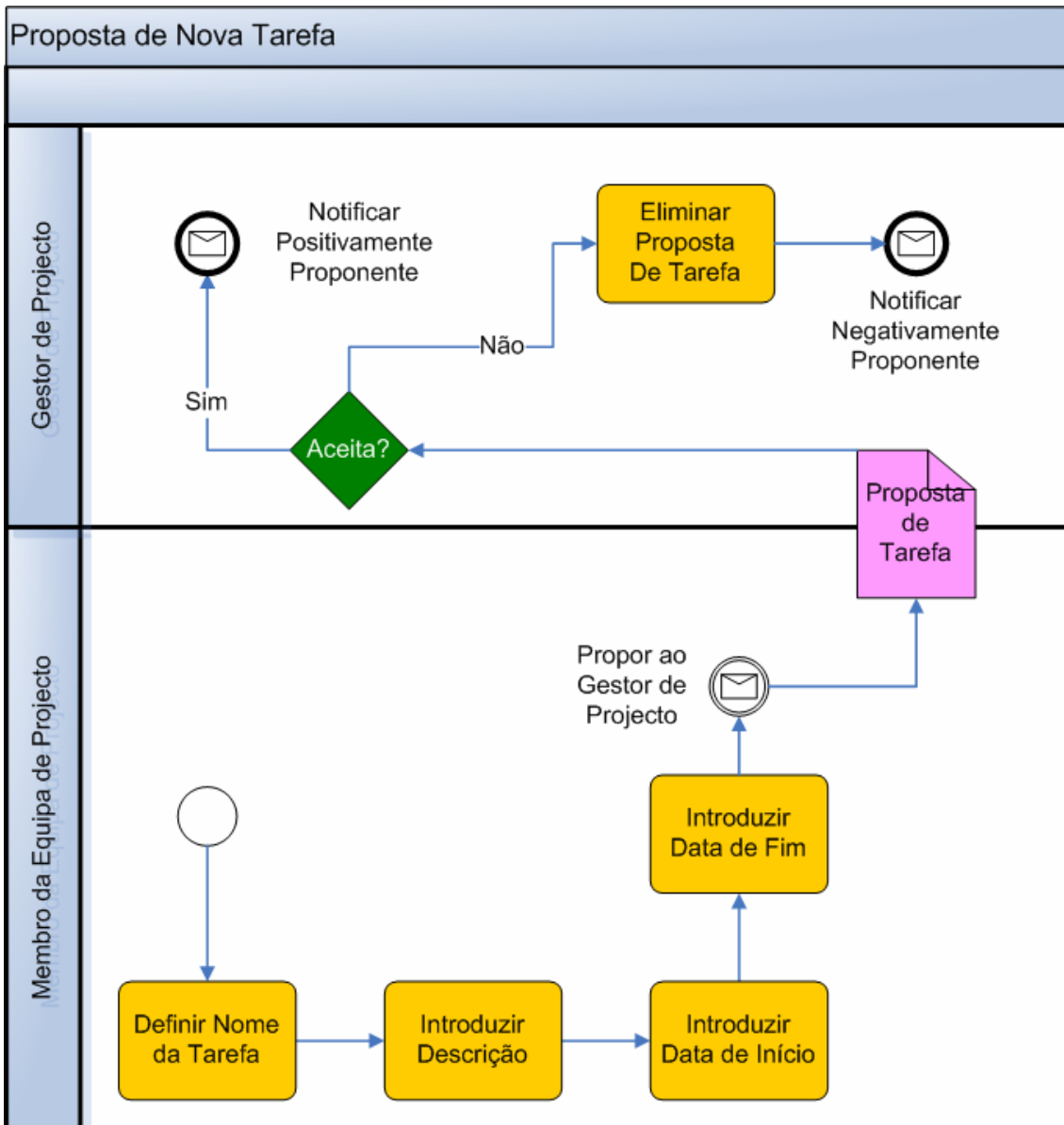


Figura 20. Criação de Nova Tarefa (Proposta de Membro da Equipa de Projecto)

4.3.4. Terminar Tarefa

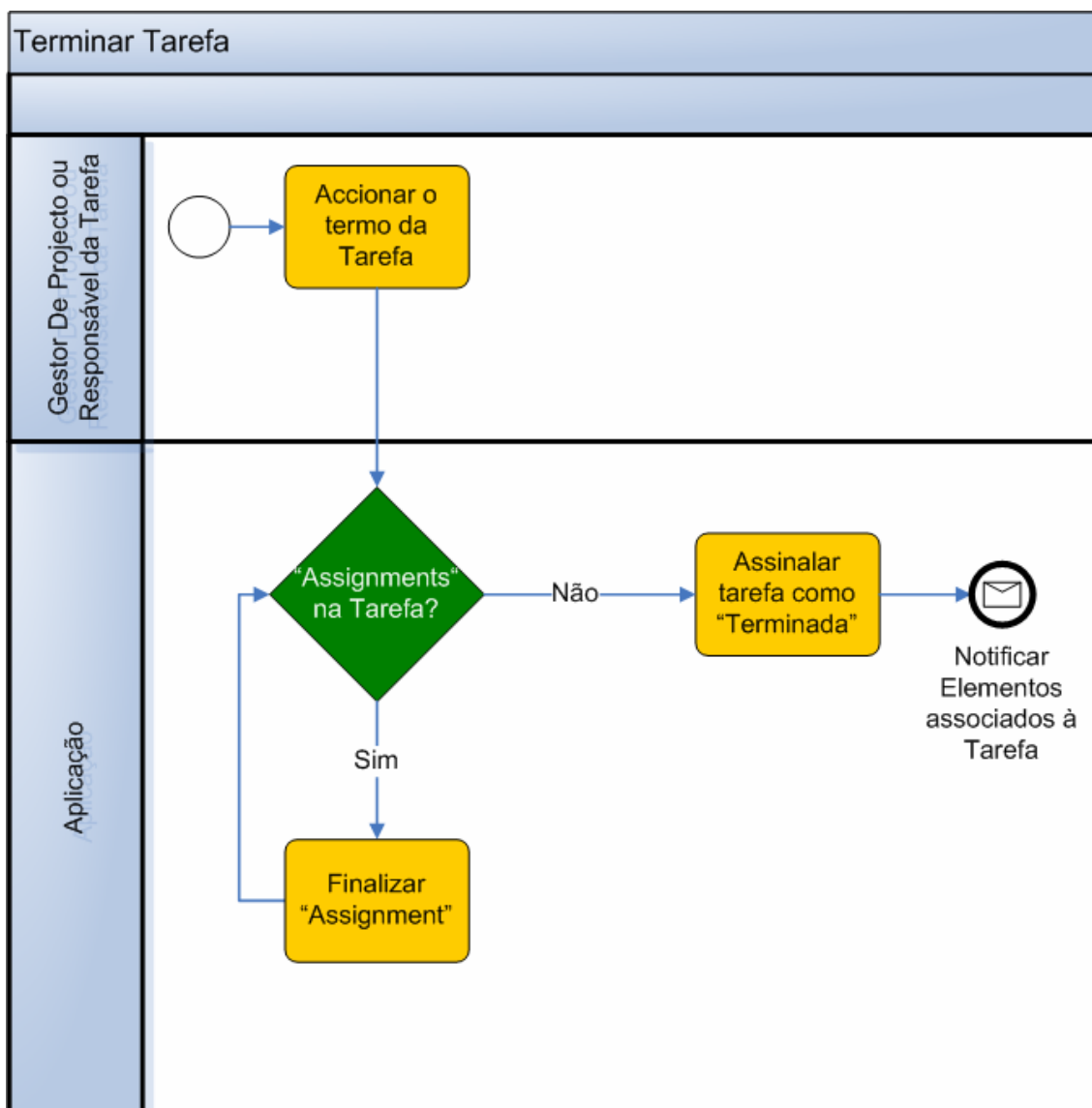


Figura 21. Terminar Tarefa

Terminar uma tarefa (Figura 21) é um sub-processo do fecho de um projecto dado que um projecto só termina quando todas as tarefas de um projecto estão terminadas. Devido à dependência do fecho de um projecto para com esta actividade, ela é igualmente crítica para manter informação coerente relativamente à disponibilidade dos recursos. Para além deste factor, a gestão de um recurso partilhado por vários projectos é realizada mesmo ao nível das tarefas por si só, pois só assim é possível otimizar a ocupação dos vários colaboradores de uma organização.

Uma tarefa também obedece a este esquema pois a uma só tarefa podem estar associados vários recursos. Do mesmo modo que um projecto só termina quando todas as

tarefas terminam, também uma tarefa só está concluída quando todos os recursos assinalarem o termo do seu trabalho na mesma.

5. Implementação

À imagem da proposta apresentada no capítulo anterior, foi desenvolvida uma ferramenta de software, seguindo o processo de desenvolvimento iterativo, para suportar a gestão de pequenos projectos que contempla todos os requisitos mencionados e implementa todos os processos propostos.

5.1. Processo de Desenvolvimento

O processo de desenvolvimento desta ferramenta seguiu uma abordagem iterativa (Figura 22) com o objectivo de reduzir os riscos logo desde início e obter feedback por parte dos utilizadores desde cedo. A implementação foi, portanto, dividida em pequenos objectivos com *milestones* a curto prazo o que permitiu uma medição do progresso mais concreta.

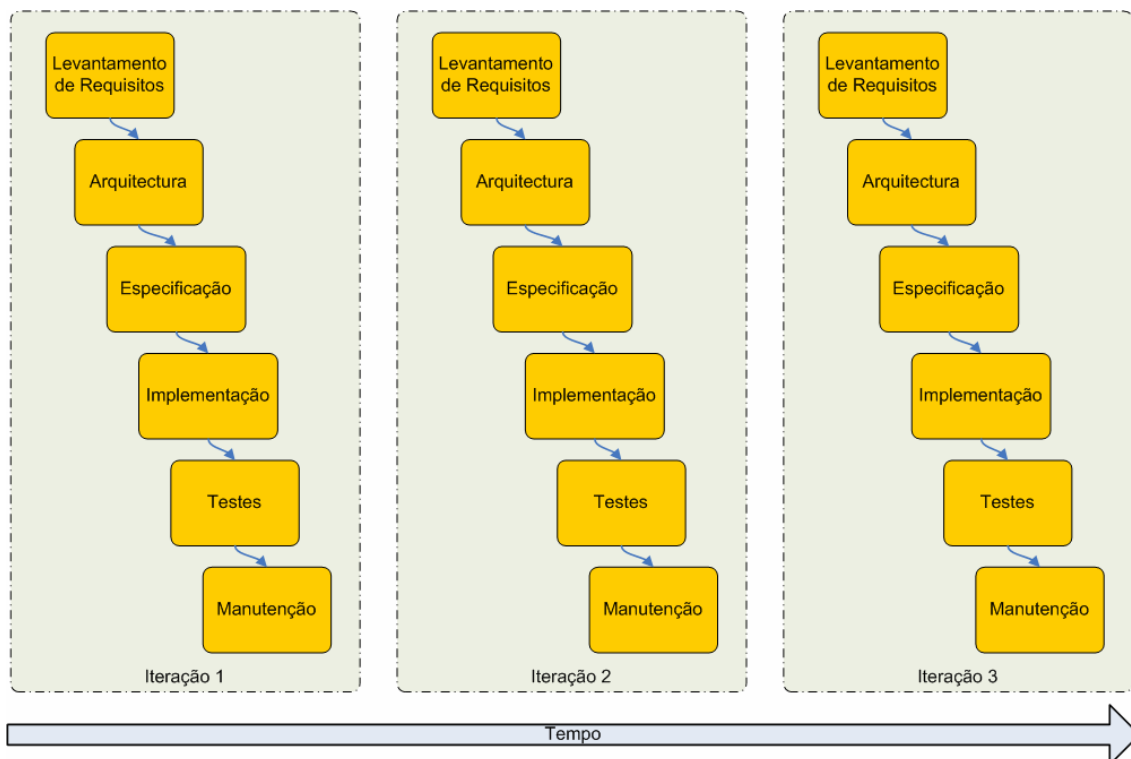


Figura 22. Processo de Desenvolvimento de Software utilizado

O projecto foi iniciado em Setembro de 2006. Até ao final de Novembro do mesmo ano foi concluída a primeira fase do processo e alguma parte da fase de Arquitectura. Porém, a lista de requisitos foi sendo completada à medida que o projecto evoluía e consoante a pesquisa efectuada.

De seguida foi realizada toda a especificação do sistema, fluxos de dados, ecrãs e funcionalidades e, conseqüentemente, a sua implementação. Esta parte do processo foi

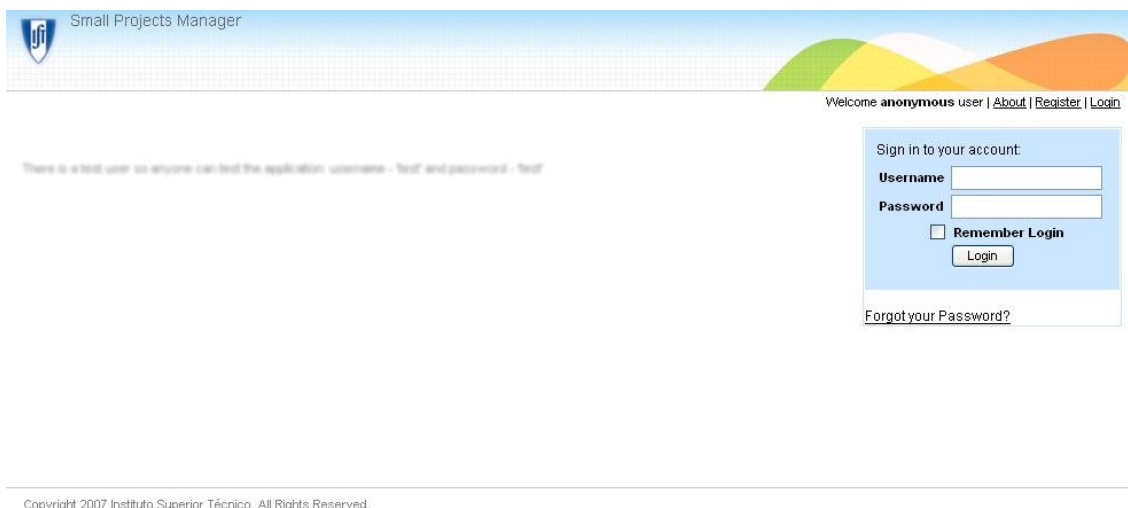
iterativa pois a implementação do sistema foi acontecendo à medida que a sua especificação evoluía. Esta fase demorou cerca de cinco meses e, no fim desse período, foi lançado primeiro protótipo.

As fases seguintes foram as fases de testes e manutenção que duraram até ao final de Junho de 2007. Neste período foram corrigidos pequenos bugs e implementadas funcionalidades que os utilizadores do sistema iam solicitando. O resultado final desta implementação foi uma aplicação *Web-based*, centralizada, independente da plataforma e com controlo de acessos.

5.2. Protótipo

A ferramenta foi desenhada com base nos critérios de análise de ferramentas apresentados na Tabela 3 e nos requisitos das metodologias apropriadas para pequenos projectos descritos no subcapítulo 4.1. A descrição que se segue explica o funcionamento da aplicação desenvolvida, bem como os requisitos que cada uma das funcionalidades apresentadas implementa.

A aplicação apresenta um mecanismo de gestão de acessos que garante diferentes privilégios. Perante o sistema existem 3 tipos de utilizadores: gestor de projecto, recurso (não gestor) e administrador. Qualquer registo poderá assumir os três papéis em simultâneo. O gestor de projecto tem privilégios que um recurso normal não apresenta. A atribuição de privilégios é efectuada assim que o utilizador se autentica na aplicação através do formulário apresentado na Figura 23:



The screenshot shows the login interface for 'Small Projects Manager'. At the top left is the logo and title. A navigation bar contains links for 'Welcome anonymous user', 'About', 'Register', and 'Login'. Below this is a message: 'There is a test user so anyone can test the application: username - test and password - test'. The main login form is titled 'Sign in to your account:' and includes fields for 'Username' and 'Password', a 'Remember Login' checkbox, and a 'Login' button. A link for 'Forgot your Password?' is located below the form. The footer contains the copyright notice: 'Copyright 2007 Instituto Superior Técnico. All Rights Reserved.'

Figura 23. Ecrã de Autenticação na Aplicação

Após a autenticação no sistema, é apresentado ao utilizador um *dashboard* onde este pode ter uma visão geral dos seus projectos, das tarefas atrasadas, das tarefas que terminam nos 7

dias seguintes, das mensagens recebidas e dos convites efectuados para integrar outros projectos. É a partir deste quadro (Figura 24 – onde está implementado parte do requisito funcional RF10 – Informação acerca dos projectos em curso, se estão atrasados ou dentro dos limites temporais) que o utilizador pode navegar através de todos os seus projectos é aceder às tarefas subsequentes.

Figura 24. Dashboard – Informação geral do utilizador (RF10)

Adicionalmente, existe no canto inferior direito uma secção com informação de administração. Esta informação apenas aparece se o utilizador tiver privilégios para tal e mostra os alertas relativos às aprovações de propostas de projectos e registos de utilizadores.

Figura 25. Proposta de novo projecto

A partir do *Dashboard* é possível aceder à funcionalidade de criação de um novo projecto. Para tal basta preencher o formulário apresentado na Figura 25:

Após o preenchimento do formulário, a proposta vai ser avaliada pelo administrador de modo a garantir que o sistema não é usado inadequadamente. Independentemente da decisão do administrador, o utilizador proponente é devidamente notificado da sua decisão.

É ainda possível, através do *Dashboard*, visualizar num gráfico Gantt (Figura 26), todos os projectos a que um utilizador está associado (como se de tarefas se tratassem). Este mecanismo confere uma vista geral da ocupação de cada recurso a um nível macro. Os requisitos funcionais RF3, RF4, RF10, RF11 e RF13 (consultar Tabela 5) estão presentes nesta funcionalidade.

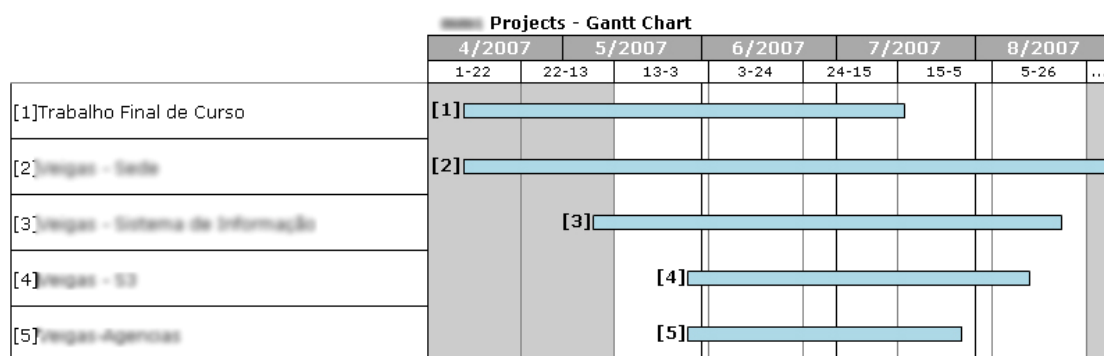


Figura 26. *Gantt Chart* de Projectos (RF3, RF4, RF10, RF11, RF13)

Através do menu principal, situado no topo da aplicação, ou do *dashboard* pode aceder-se à página de “caixa de entrada”, onde são guardadas as mensagens trocadas entre os utilizadores, e à página do “calendário pessoal”, a partir do qual o utilizador pode consultar as suas tarefas.

O ecrã de mensagens (Figura 27) permite aos utilizadores gerirem as mensagens escritas recebidas no âmbito dos projectos de que faz parte. Uma mensagem está sempre associada a um projecto e, apenas a partir do ecrã geral de cada projecto, é que os elementos de uma equipa de projecto podem trocar mensagens entre si. Este mecanismo centraliza os meios de comunicação na própria aplicação evitando a utilização do e-mail em várias situações e, como tal, preenche o requisito funcional RF8 – Proporcionar meios de comunicação centralizados e que proporcionem um ambiente colaborativo.



Figura 27. Ecrã de Gestão de Mensagens (RF8)

O ecrã de calendário (Figura 28) é o ponto de cruzamento de todos os projectos em que um utilizador está envolvido. Neste ecrã está disponível um conjunto de filtros que, quando manipulados, geram diferentes vistas respeitantes às tarefas comuns a cada utilizador. O objectivo principal é ajudar cada um dos utilizadores a gerir o seu próprio tempo e o dos elementos das equipas dos projectos que gere, com base nas tarefas que cada um deles tem associadas num determinado dia ou período. Para tal, foram incluídos nesta funcionalidade filtros temporais (dia específico, semanal, mensal e global), filtro de *status* de modo a filtrar as tarefas já executadas ou atrasadas, e um filtro de responsabilidade. Este último serve os gestores ao apresentar, não só as suas próprias tarefas, mas também as tarefas associadas a outros recursos dos projectos que ele está a gerir de modo a perceber se existem conflitos com esses recursos.

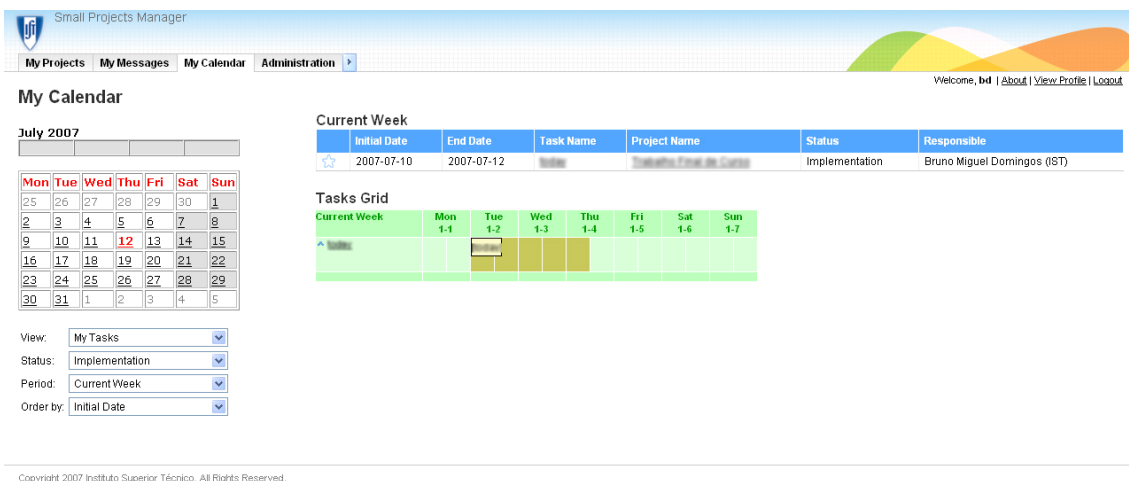


Figura 28. Calendário de tarefas individual para cada utilizador (RF1, RF10, RF11, RF12, RF13)

É visível através da Figura 28 que também existem diferentes modos de representação gráfica dois dados resultantes da filtragem seleccionada. Os modelos de visualização

disponíveis são o modo tabela e modo grelha (com período semanal). Deste modo, o controlo semanal das tarefas ganha outro ênfase e torna a tarefa mais simples para o gestor. Estas funcionalidades vão de encontro aos requisitos funcionais RF1, RF10, RF11, RF12 e RF13 enunciados na Tabela 5.

The screenshot shows the 'Small Projects Manager' interface. At the top, there are navigation tabs: 'My Projects', 'My Messages', 'My Calendar', and 'Administration'. Below this, the project name 'Project 'Trabalho Final de Curso'' is displayed, along with its 'Current Status: Implementation'. There are links for 'Export Project (MS Project format)', 'Import Tasks', 'Gantt Chart', and 'Plan Grid'. A 'Mark Project As Finished' button is visible. Below the navigation, there are tabs for 'Tasks', 'Risks', 'Issues', 'Files', 'Requirements', 'Tasks Suggestions', 'Team', 'Deliverables', 'Notes', and 'Forums'. The 'Tasks' tab is active, showing a table of tasks. The table has columns for 'Initial Date', 'End Date', 'ID', 'Task Name', 'Responsible', 'Status', 'Predecessors', and 'Edit'. The tasks listed are:

Initial Date	End Date	ID	Task Name	Responsible	Status	Predecessors	Edit
2007-04-09	2007-05-11	1	Melhoria do SPM (Melhoria do SPM)	Bruno Miguel Domingos (IST)	Finished		-
2007-05-14	2007-05-11	2	Melhoria do SPM terminada (Melhoria do SPM terminada e fechada)	Bruno Miguel Domingos (IST)	Finished	1	-
2007-05-07 Place: [redacted] Time: 17:30:00	2007-05-08	3	3ª Reunião	Bruno Miguel Domingos (IST)	Finished 1 day(s) overdue!		-
2007-07-16	2007-07-16	4	Relatório Final terminado (Relatório Final terminado)	Bruno Miguel Domingos (IST)	Implementation		[edit icon]
2007-07-10	2007-07-12	5	[redacted]	Bruno Miguel Domingos (IST)	Implementation		[edit icon]

At the bottom of the page, there is a copyright notice: 'Copyright 2007 Instituto Superior Técnico. All Rights Reserved.'

Figura 29. Vista geral de um projecto (RF1, RF3, RF5, RF7, RF9, RF10, RF11)

A Figura 29 ilustra o ecrã principal de um projecto. Dada importância deste ecrã, um vasto número de requisitos estão aqui representados - RF1, RF3, RF5, RF7, RF9, RF10 e RF11 – ver Tabela 5. É a partir dele que o gestor de projecto e os restantes elementos da equipa podem aceder a toda a informação relativa às tarefas, aos riscos, *issues*, ficheiros relativos ao projecto, requisitos, *deliverables*, notas e fóruns.

Como foi referido, o gestor do projecto tem acesso a determinadas funcionalidades que os restantes elementos da equipa de projecto não têm, p. ex., criar / apagar tarefas, dar um projecto como terminado, acrescentar riscos ao projecto e criar a equipa de projecto. Os restantes elementos da equipa de projecto estão um pouco mais limitados, mas existem métodos alternativos para estes participarem no planeamento e controlo do projecto.

Por exemplo, com o objectivo de promover a colaboração de toda a equipa no planeamento do projecto, existe a possibilidade de um elemento da equipa propor a inclusão de uma tarefa no projecto, bastando para isso que este preencha um formulário com a descrição da mesma. A diferença é que na altura em que o gestor de projecto cria uma tarefa, esta fica automaticamente disponível, enquanto se for um outro elemento da equipa, a proposta da tarefa submetida passa pela aprovação do gestor através de um processo rápido

onde este apenas necessita de assinalar num formulário a sua decisão – aprovação ou rejeição.

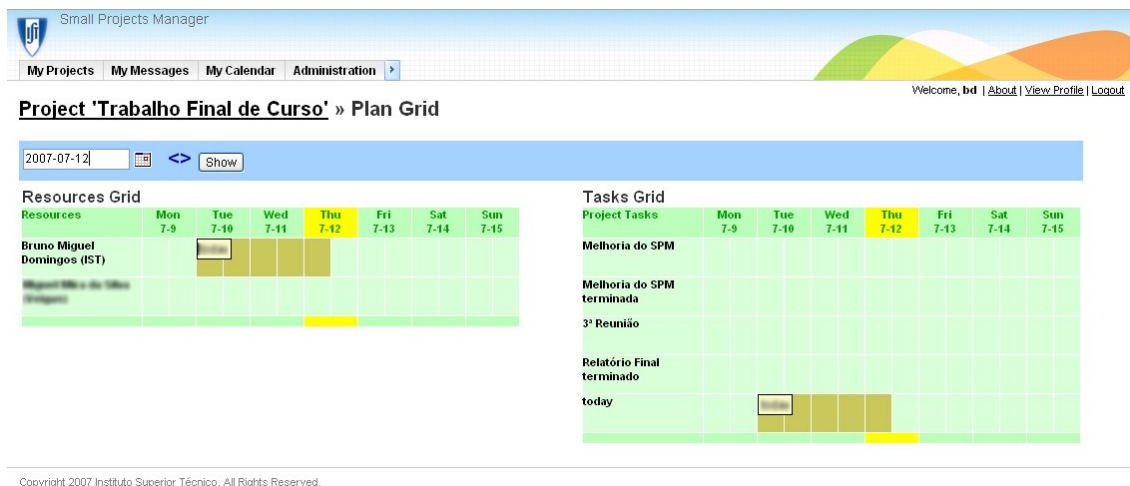


Figura 30. Grelha de tarefas de um projecto (RF1, RF4, RF10, RF11, RF13)

Para além das funcionalidades descritas, existe um outro conjunto acessível a todos os elementos da equipa de projecto. Qualquer elemento da equipa de projecto pode consultar as tarefas que compõem o projecto com os respectivos *deadlines* e *deliverables*, exportar o projecto segundo a formatação do MS Project, anexar novos ficheiros ou *issues* ao projecto e ainda visualizar o gráfico *Gantt* do projecto ou a grelha de tarefas (Figura 30). Esta vista dos dados mostra simultaneamente a organização temporal das tarefas e a alocação de recursos nas mesmas. É uma mais-valia para qualquer elemento da equipa, mas sobretudo para o gestor que vê facilitada a tarefa de planeamento e controlo. Os requisitos aqui implementados são os RF1, RF4, RF10, RF11 e RF13 e estão descritos na Tabela 5.

Esta ferramenta implementa também os requisitos de gestão de riscos e *issues* mencionados no capítulo 4.2. Como tal, apresenta um mecanismo de gestão dos mesmos (Figura 31), obedecendo deste modo ao requisito funcional RF6 – Fornecer meios para a gestão de riscos e *issues*. Toda a equipa de projecto pode acompanhar de modo simples e directo a identificação de novos riscos, bem como a sua actualização e ciclo de vida, ou seja, a sua resolução e estado. Apenas o gestor tem permissões para incluir novos riscos no projecto mas, através dos mecanismos de comunicação existentes, torna-se fácil a qualquer elemento da equipa colaborar na identificação de novos riscos.

Small Projects Manager

My Projects My Messages My Calendar Administration

Project 'Trabalho Final de Curso'

Current Status: Implementation

Export Project (MS Project format) | Import Tasks | Gantt Chart | Plan Grid

Mark Project As Finished

Tasks Risks Issues Files Requirements Tasks Suggestions Team Deliverables Notes Forums

Project Risks

New Risk

Definition	Resolution	Status	Priority	Identified on	Edit	Delete
Atraso na dissertação	Aumentar o número de horas diárias de escrita	Open	Medium	2007-07-05		

Copyright 2007 Instituto Superior Técnico. All Rights Reserved.

Figura 31. Mecanismo de gestão de riscos (RF6)

Para a gestão de *issues* foram igualmente criadas funcionalidades para o seu acompanhamento. Todos os elementos da equipa de projecto podem aceder ao ecrã representado na Figura 32 e adicionar novas questões que considerem pertinentes. Um *issue* é representado pela sua descrição, o potencial impacto no projecto, data de identificação, estado, o responsável pelo seu fecho e o risco associado.

Small Projects Manager

My Projects My Messages My Calendar Administration

Project 'Trabalho Final de Curso'

Current Status: Implementation

Export Project (MS Project format) | Import Tasks | Gantt Chart | Plan Grid

Mark Project As Finished

Tasks Risks Issues Files Requirements Tasks Suggestions Team Deliverables Notes Forums

Project Issues

New Issue

Description	Potential Impact	Status	Open Date	End Date	Responsible	Risk	Edit	Delete
Falta de dados primários capitais	3	Open	2007-06-25	2007-07-10	Bruno Miguel Domingos (IST)	Atraso na dissertação		

Copyright 2007 Instituto Superior Técnico. All Rights Reserved.

Figura 32. Mecanismo de gestão de *issues* (RF6)

Sempre que um *issue* é adicionado ao projecto é simultaneamente criada uma entrada nos fóruns do projecto (Figura 33) com o objectivo de promover o debate entre os elementos da equipa. Deste modo, torna-se mais fácil, quer a comunicação entre a equipa, quer a resolução das questões pendentes.

Todavia, o objectivo dos fóruns não se cinge apenas à discussão dos *issues*. O objectivo principal deste mecanismo é fortalecer os canais de comunicação entre a equipa, ajudando e motivando o diálogo mesmo entre os elementos com mais dificuldades em expressar-se. Adicionalmente, os fóruns são locais à ferramenta e, como tal, o acesso é rápido, automático e centralizado, assegurando-se mais uma vez a implementação do requisito funcional RF8 – Proporcionar meios de comunicação centralizados e que proporcionem um ambiente colaborativo.

Small Projects Manager

My Projects My Messages My Calendar Administration

Project 'Trabalho Final de Curso'

Current Status: Implementation

Export Project (MS Project format) | Import Tasks | Gantt Chart | Plan Grid

Mark Project As Finished

Tasks Risks Issues Files Requirements Tasks Suggestions Team Deliverables Notes Forums

Project Forums

New Forum

Author	Title	Date	Coment	Download	Remove Attach	Coment	Edit	Delete
bd	teste	2007-04-19 12:28:23	#	Download	X	Coment	✎	X
bd	teste...Resumo dos dois primeiros capitulos...	2007-07-05 00:00:00	Resumo dos dois primeiros capitulos	Attach File	-	Coment	✎	X

Copyright 2007 Instituto Superior Técnico. All Rights Reserved.

Figura 33. Fóruns do projecto (RF8)

Outra forma de comunicação existente na aplicação é o já referido mecanismo de mensagens. Este mecanismo está disponível a partir do ecrã apresentado na Figura 34 referente à página de gestão da equipa de projecto.

Small Projects Manager

My Projects My Messages My Calendar Administration

Project 'Trabalho Final de Curso'

Current Status: Implementation

Export Project (MS Project format) | Import Tasks | Gantt Chart | Plan Grid

Mark Project As Finished

Tasks Risks Issues Files Requirements Tasks Suggestions Team Deliverables Notes Forums

Project Team

Add New Member

Name	Username	E-mail	Remove from Team
Bruno Miguel Domingos (IST) (Manager)	bd	domingos.bruno@gmail.com	-
Miguel Melo de Sá (Programador)	melo	melmelo@ist.utl.pt	X

Groups

Add New Group

Group Name	User Name
grupo de sistemas de informação	Bruno Miguel Domingos (IST)
	Miguel Melo de Sá (Programador)

Copyright 2007 Instituto Superior Técnico. All Rights Reserved.

Figura 34. Equipa de Projecto

Parte das funcionalidades aqui apresentadas encontram-se, mais uma vez, limitadas ao gestor de projecto. Este tem a possibilidade de adicionar e remover elementos da equipa de projecto, e de criar grupos de utilizadores. Acessíveis a toda a equipa estão os dados dos vários elementos da equipa e a possibilidade de troca de mensagens no contexto do projecto em causa.

The screenshot shows the 'Small Projects Manager' web application. At the top, there is a navigation menu with 'My Projects', 'My Messages', 'My Calendar', and 'Administration'. The main content area displays details for a task titled 'Project 'Trabalho Final de Curso' » Task '...'. The 'Current Status' is 'Implementation'. A 'Task is on schedule.' message is shown. The 'Responsible User' is 'Bruno Miguel Domingos (IST)'. There is a 'Mark Task as Finished' button. The task dates are 'Initial Date: 2007-07-10' and 'End Date: 2007-07-12'. Below this, there is an 'Edit Task' link and a set of tabs: 'Human Resources', 'Assets', 'Deliverables', 'Predecessors', 'Dependent Tasks', 'Requirements', and 'Notes'. The 'Human Resources (Assignments)' tab is active, showing a table with one entry for 'Bruno Miguel Domingos (IST)' and a 'Mark As Finished' button. A 'Delete' link is also present. The footer contains the copyright notice: 'Copyright 2007 Instituto Superior Técnico. All Rights Reserved.'

Figura 35. Ecrã geral de uma tarefa (RF1, RF3, RF5, RF7, RF11)

A Figura 35 ilustra o ecrã onde é possível visualizar todos os detalhes de uma tarefa. Analogamente ao ecrã principal de um projecto, também este ecrã é de extrema importância dado que toda a informação relativa a uma dada tarefa é acessível através dele. Como tal, um número significativo de requisitos estão presentes neste ecrã (RF1, RF3, RF5, RF7 e RF11) o qual permite associar tarefas entre si através de dependências do tipo *Finish-to-Finish*, *Finish-to-Start* e *Start-to-Start*, alocar recursos tendo por base os elementos pertencentes à equipa de projecto, associar *deliverables* e requisitos que a tarefa tenha que cumprir e exportar a tarefa para formato *iCalendar* (MS Outlook) de modo a possibilitar a integração com as aplicações mais comuns de gestão de calendário. Mais uma vez, existem acções que só o gestor de projecto ou o responsável pela execução da tarefa podem tomar, nomeadamente marcar a tarefa como terminada ou alocar os recursos.

A partir do momento que um recurso é alocado a uma tarefa, fica disponível um mecanismo a partir do qual o este pode e deve assinalar a conclusão do seu trabalho na tarefa, sempre que tal se verifique. Também esta funcionalidade está protegida por controlo de acessos.

Os requisitos listados no subcapítulo 4.2.2 referentes à gestão de *deliverables* estão igualmente implementados nesta aplicação. Estes requisitos são de extrema importância dado que os *deliverables* são as caras dos projectos, são o produto que chega ao cliente e, como tal, é necessário cumprir os prazos de entrega dos mesmos. Para auxiliar as equipas de projecto a cumprir com os *deadlines* a aplicação apresenta algumas funções de gestão que dão suporte aos *deliverables* de cada projecto. Qualquer elemento da equipa de projecto pode adicionar *deliverables*, assinalar a entrega, associá-los a tarefas e anexar ficheiros importantes aos mesmos ou, no caso de o próprio *deliverable* ser um relatório, anexá-lo.

A Figura 36 representa os dois ecrãs associados à gestão de *deliverables*. O ecrã de fundo é acessível através do ecrã geral de um projecto (Figura 29) e apenas permite visualizar o estado dos *deliverables* e navegar entre as actividades a eles associadas.

The image contains two screenshots of the Small Projects Manager web application. The left screenshot shows the 'Project Deliverables' section for a project named 'Trabalho Final de Curso'. It includes a navigation menu with 'My Projects', 'My Messages', 'My Calendar', and 'Administration'. Below the project name, there are tabs for 'Tasks', 'Risks', 'Issues', 'Files', 'Requirements', 'Tasks Suggestions', 'Team', and 'Deliverables'. A table lists deliverables with columns for 'Task Name', 'Deliverable', 'Status', 'Deliver Date', and 'Attach'. The right screenshot shows the 'Task Deliverables' section for a specific task. It includes a navigation menu and tabs for 'Human Resources', 'Assets', 'Deliverables', 'Predecessors', 'Dependent Tasks', 'Requirements', and 'Notes'. A table lists 'New Deliverable' with columns for 'Description', 'Status', 'Deliver Date', 'Attach', 'Edit', and 'Delete'.

Figura 36. Mecanismo de gestão de *deliverables* (RF5)

O ecrã mais próximo é acessível através do ecrã geral de uma tarefa (Figura 35) e nele é possível associar à tarefa em causa o(s) *deliverable*(s) que lhe dizem respeito. É também a partir deste ecrã que é possível assinalar o estado do *deliverable* e anexar ficheiros. O requisito funcional RF5 – Assinalar marcos e entregas – é implementado por este mecanismo na sua totalidade.

Esta ferramenta foi também desenvolvida dando especial atenção ao factor tempo que em pequenos projectos é, tipicamente, curto. Dado que os utilizadores de ferramentas como esta passam, logicamente, o seu tempo na implementação dos projectos, esta ferramenta contempla também um motor de notificações sempre que algo relevante aconteça no projecto.



The task [redacted] of the Project Trabalho Final de Curso ends today!
To 'Mark Task as Finished' press the link below.
For detailed information access the application!

Click to fast access: [http://\[redacted\]/SPM/Task.aspx?ID=4&ID_Task=264&ProjectName=Trabalho Final de Curso&EndTask=endtask](http://[redacted]/SPM/Task.aspx?ID=4&ID_Task=264&ProjectName=Trabalho Final de Curso&EndTask=endtask)

You can find us at: [http://\[redacted\]/SPM](http://[redacted]/SPM)

Figura 37. Exemplo de notificação enviada pelo sistema (RF14)

A Figura 37 ilustra um exemplo de um e-mail enviado pelo sistema quando uma actividade está prestes a terminar segundo o planeamento do projecto. Como esta, existem outras notificações que informam as equipas de projectos sempre que um novo risco é associado ao projecto, um *deliverable* é marcado ou uma tarefa está atrasada. O requisito funcional RF14 é deste modo implementado.

O já referido papel de administrador do sistema é análogo ao de um moderador de uma conferência. No caso concreto desta aplicação, este fica a cargo de:

- avaliar as propostas de projectos efectuadas pelos utilizadores da ferramenta (Figura 38)

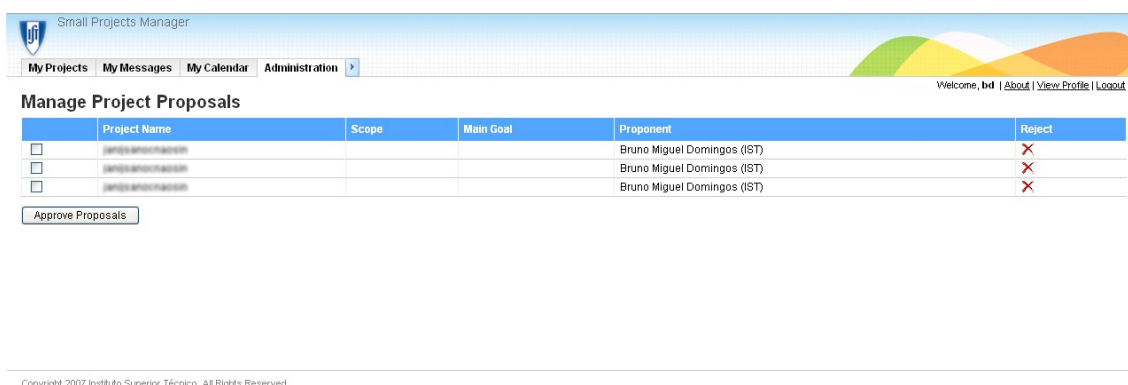


Figura 38. Gestão de propostas de projectos (RF2)

- administrar os projectos que estão atribuídos a gestores (Figura 39)

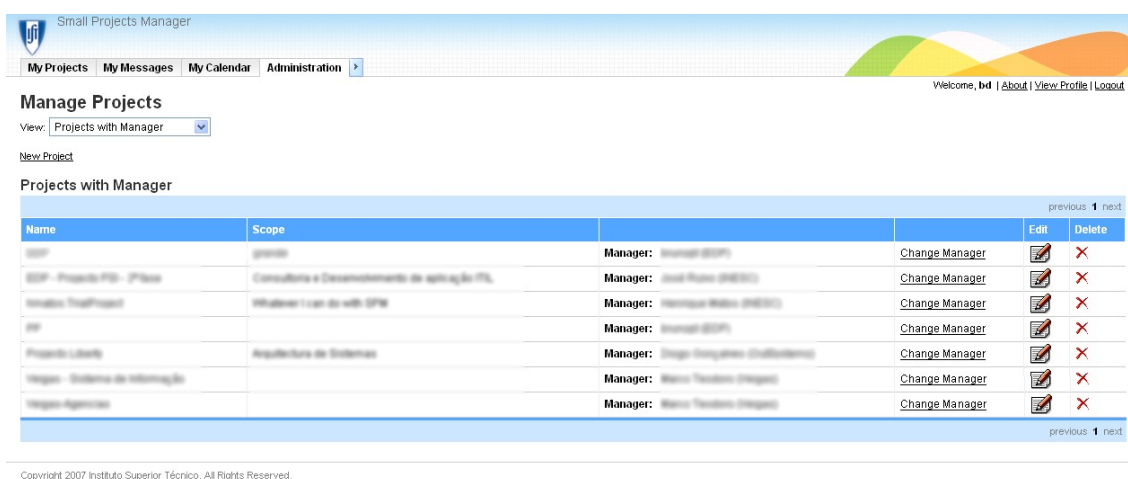


Figura 39. Gestão de projectos que estão a ser geridos com auxílio da ferramenta (RF2)

- gerir os dados e permissões dos próprios utilizadores da ferramenta (Figura 40)

Small Projects Manager

My Projects My Messages My Calendar Administration

Welcome, **Id** | [About](#) | [View Profile](#) | [Logout](#)

Manage Users

[New User](#)

View: All Users From: All Units

previous 1 2 next

Name	Unit	Username	Email	Approved	Administrator	Edit	Delete
andré	IST	andré	administrator@igquery.org	✓	✗		
André Agreia	Trendis	agreia	andres.agreia@trendis.com	✓	✗		
André Rocha	IS-Complex	ar	arocha@is-complex.pt	✓	✗		
Bruno	INESC	b_kamari	b.kamari@inesc.com	✓	✗		
Bruno	OutSystems	bruno	bruno.fernandes.p@gmail.com	✓	✗		
Bruno Miguel Domingos	IST	bd	brunomiguel.domingos@gmail.com	✓	✓		
Bruno Rodrigues	Vegges	br	bruno.rodrigues@gmail.com	✓	✗		
Bruno	EDP	brunopt	bruno.fernandes.p@gmail.com	✓	✗		
Carma	INESC	ccardin	carma.cardin@gmail.com	✓	✗		
Diogo Gonçalves	OutSystems	dg	diogo.goncalves@terterlogica.pt	✓	✗		
Eduardo Cruz	OutSystems	emc	eduardo.cruz@outsystems.com	✓	✗		
João Maria	Karna	jmara	j.mara@karna-consulting.pt	✓	✗		
José Carlos Delgado	IST	jc	joscar.delgado@ist.utl.pt	✓	✗		
José Carlos Santos	Vegges	js	jsantos@gmail.com	✓	✗		

previous 1 2 next

Copyright 2007 Instituto Superior Técnico. All Rights Reserved.

Figura 40. Gestão dos dados dos utilizadores da aplicação

Este protótipo encontra-se ainda em utilização e em constante evolução. No capítulo seguinte é efectuada a avaliação da sua utilização num caso prático onde foram avaliados principais objectivos desta tese e mais especificamente da ferramenta.

6. Avaliação

Depois da fase de implementação e desenvolvimento da ferramenta apresentada é necessário avaliá-la e perceber se os resultados obtidos vão de encontro aos inicialmente previstos. De forma a comparar a ferramenta desenvolvida com as restantes avaliadas no capítulo 2.3.2 deste documento, estão representadas na Tabela 6 os requisitos implementados pela mesma.

Tabela 6. Avaliação da proposta por comparação com as ferramentas analisadas

	MS Project	Quickbase	Basecamp	Protótipo Proposto
Facilidade de Utilização	●●●	●●	●●●●●	●●●●●
Não necessita de treino/especialização	●	●●	●●●●●	●●●●●
Funcionalidades de planeamento e controlo	●●●●●	●●●●●	●●	●●●
Gestão de recursos	●●●●	●●●●	●	●●●
Gestão de riscos	●●	●●●	●	●●●●
Gestão de requisitos	●	●	●	●●●●●
Espaço para documentação	●	●●●●	●	●●●●
Suporte à comunicação	●	●	●●●●●	●●●●●

(● – Inexistente; ●● – Fraco; ●●● – Médio; ●●●● – Bom; ●●●●● – Muito Bom)

Através dos dados da Tabela 6 pode verificar-se que existem requisitos que não satisfazem ainda os níveis pretendidos. Apesar de não ser o objectivo principal da ferramenta, ao nível das funcionalidades de planeamento e controlo, a ferramenta desenvolvida está longe de ser tão completa como o MS Project. No entanto este valor não é tão significativo como o do requisito de gestão de recursos. Como foi referido ao longo deste documento, a gestão de recursos em ambientes multi-projecto tem contribuído imenso para o insucesso dos projectos de SI. Como

tal, um mecanismo que garanta a gestão completa dos recursos cruzando os dados dos vários projectos de uma organização é, certamente, uma funcionalidade essencial.

Outros pontos como a gestão de riscos e documentação ficaram ligeiramente abaixo do valor máximo. Contudo, os objectivos associados a estes requisitos foram alcançados, pois os resultados obtidos a partir da utilização da ferramenta neste âmbito foram satisfatórios não havendo, por parte dos utilizadores, grandes dificuldades de utilização ou necessidade de funcionalidades extra.

Relativamente à gestão de requisitos, devido à pouca utilização das funcionalidades implementadas, não foram apresentadas propostas. Como tal, pode concluir-se que, apesar da carência de dados, os mecanismos apresentados, preenchem as necessidades dos utilizadores.

A facilidade de utilização da ferramenta foi sendo cada vez maior consoante a utilização prática do protótipo. Inicialmente, mesmo seguidas as regras de usabilidade mais conhecidas, a falta de interacção com utilizadores comprometeu um pouco este requisito que veio a ser melhorado de forma progressiva no decorrer do caso de estudo. Por fim, os novos utilizadores não sentiam dificuldades em utilizar a ferramenta e, conseqüentemente, dispensaram qualquer treino ou especialização.

No âmbito do suporte à comunicação, a utilização da ferramenta apresentou-se com bons resultados. A receptividade dos utilizadores foi positiva e houve propostas para a implementação de outros mecanismos de suporte por eles sugeridos. O caso mais relevante foi o da inclusão de mensagens escritas associadas aos projectos.

De seguida encontra-se uma nota sobre a tecnologia utilizada, a descrição de todos os protótipos e respectivos testes, bem como uma análise das entidades informacionais que passaram ou ainda se encontram nesses protótipos.

6.1. Tecnologia

O sistema apresentado foi desenvolvido sobre a plataforma OutSystems Hub Edition. Esta tecnologia segue a uma arquitectura *three-tier* para o desenvolvimento de aplicações Web. A Figura 41 ilustra esta plataforma:

- Service Studio: Ferramenta *drag-and-drop* de desenvolvimento.
- Service Center: Aplicação Web de administração do servidor onde estão alojadas as aplicações.
- Integration Studio: Ferramenta de integração cujo resultado são componentes OutSystems.

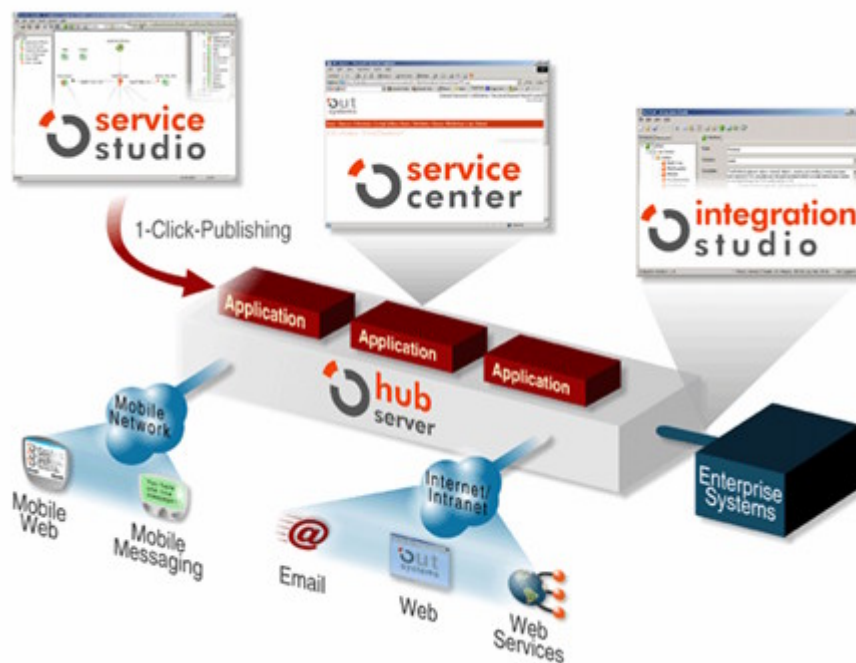


Figura 41. Plataforma tecnológica da Arquitectura OutSystems

Vários factores contribuíram para a utilização desta plataforma, dentro dos quais, o facto do período de desenvolvimento ser curto, a curva de aprendizagem ser pouco acentuada e o produto resultante desta ser uma aplicação *Web-based*. Este último foi um ponto de reflexão dado que o impacto no produto final é elevado.

Apesar de tudo, decidiu-se utilizar esta abordagem pois ela iria de encontro a um requisito fundamental: a centralização de toda a informação do projecto acessível a todos e independente de sistema operativo. Adicionalmente, a Web é, hoje em dia, a melhor forma de criar ambientes colaborativos quando os recursos envolvidos se encontram geograficamente distantes.

6.2. 1º Protótipo

A primeira experiência de utilização desta ferramenta teve início do mês de Março de 2007 e aconteceu numa empresa do ramo da imobiliária, no âmbito de vários projectos ligados à reestruturação da infra-estrutura informática da mesma.

Este foi o cenário ideal para começar a testar a aplicação pois existiam vários recursos desmembrados a contribuir para vários dos pequenos projectos que estavam a acontecer em paralelo. O empenho foi mútuo pois havia interesse por parte dos colaboradores da empresa em serem auxiliados por uma ferramenta que respondesse às necessidades que eles sentiam. Por outro lado, havia interesse em testar a aplicação e perceber quais as falhas que o sistema apresentava.

Os objectivos definidos para esta fase foram:

- Testar as funcionalidades implementadas
- Perceber a receptividade dos utilizadores a uma ferramenta de gestão de projectos *Web-based*
- Testar a usabilidade da aplicação dado que até esta fase esta foi desenvolvida apenas com algumas contribuições de alguns potenciais utilizadores
- Levantar necessidades ligadas com algumas funcionalidades e modos de apresentação da informação
- Explorar a tecnologia e perceber em que pontos esta poderia ser ainda mais útil ao conceito da aplicação.

Esta actividade teve uma duração de 2 meses durante os quais houve feedback semanal, por vezes, mais do que uma vez por semana. Este feedback relacionava-se essencialmente com pedidos de alteração da interface e execução de tarefas, e pedidos de novas funcionalidades. Alguns dos pedidos mais importantes registados foram:

- Criação de filtros de estado e ordenações nas tarefas de forma a reduzir o número de tarefas na tabela principal e facilitar a pesquisa daquelas que o utilizador pretende
- Fornecimento de meios gráficos (tipo Gráfico Gantt) de forma a facilitar o esquema mental em relação à distribuição temporal das tarefas no projecto
- Fornecimento de alertas aos utilizadores sempre que uma actividade é criada e este lhe é alocado
- Manter informação útil no *dashboard*, mais concretamente as tarefas que terminam nos 7 dias seguintes
- Nos e-mails de notificação, acrescentar hiperligações de modo a agilizar a acessibilidade aos conteúdos em questão
- Aproximação de um Gráfico Gantt substituindo as tarefas por projectos, ou seja, uma perspectiva temporal de todos os projectos onde o utilizador está alocado
- Criação de um mecanismo de mensagens de modo a fortalecer o ambiente colaborativo do sistema e, ao mesmo tempo, reduzir o número de notificações por e-mail
- Mecanismo de gestão de *issues*.

O acompanhamento e a fase de alterações terminaram quando o sistema chegou a um ponto de equilíbrio em que os colaboradores estavam satisfeitos e o sistema tinha atingido uma completude já considerável.

Mesmo depois de este processo terminar, a empresa continuou a gerir os projectos de sistemas de informação e infra-estruturas tecnológicas a partir desta aplicação.

6.3. 2º Protótipo

O segundo protótipo foi lançado no dia 29 de Abril de 2007, dias depois da fase de testes e pedidos evolutivos referida no subcapítulo anterior ter sido concluída.

Nesta versão houve um pouco mais de ambição. A grande aposta foi publicar a solução no portal da *extranet* OutSystems de modo a que todos os seus clientes fossem notificados e ficassem a conhecer a aplicação. A responsabilidade foi maior do que no teste anterior pois esta comunidade reúne clientes importantes e, como tal, de maior exigência.

Rapidamente aumentaram o número de utilizadores e projectos. O sistema foi realmente sujeito a cargas reais e a projectos de outros negócios que lhe conferiram uma maior versatilidade, tendo assim conseguido cumprir os objectivos definidos para esta fase.

Registaram-se ainda alguns pedidos de funcionalidades interessantes que vieram a ser implementados durante este período. O caso mais evidente e enriquecedor foi a integração com uma outra solução OutSystems que viera encaixar na perfeição com a aplicação, o *Plan Grid* que já foi explicado detalhadamente no subcapítulo 5.2.

A Figura 42 mostra um exemplo da vista semanal proporcionada pelo *Plan Grid* de um pequeno projecto gerido na com o auxílio da aplicação:

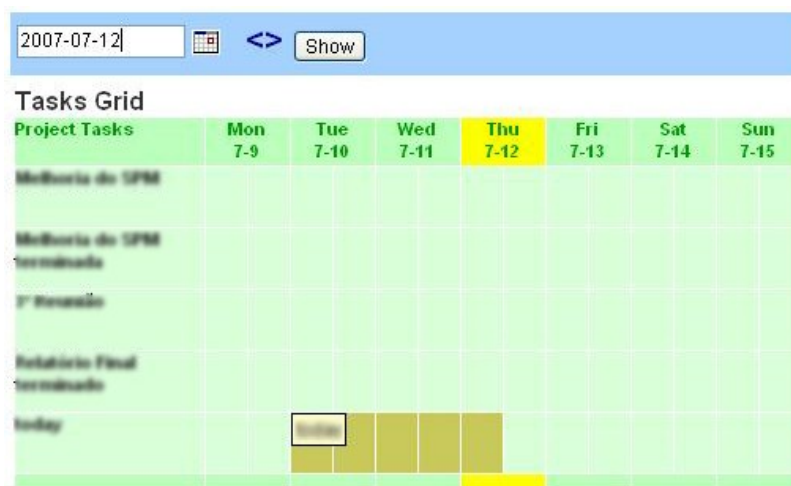


Figura 42. Vista semanal do Plan Grid

6.4. Conclusão

Durante 6 meses a aplicação foi testada por 30 utilizadores que geriram ou fizeram parte da equipa de 17 projectos. Destes 17 projectos 11 estão já concluídos, sendo possível consultar o histórico dos mesmos através da aplicação, e os restantes 6 projectos encontram-se a ser geridos ainda com suporte da ferramenta proposta.

Os objectivos propostos em cada uma destas duas fases foram atingidos com sucesso. O feedback fornecido pelos utilizadores foi crucial para este sucesso e conferiu à aplicação uma maior credibilidade e segurança para futuros utilizadores.

7. Conclusão

O objectivo principal desta tese é contribuir para a resolução dos problemas que contribuem para o insucesso de pequenos projectos de sistemas de informação. Os problemas identificados a partir de casos concretos relatados na literatura foram agrupados segundo três grupos principais: metodologias e cultura das organizações, ferramentas de SW utilizadas e características dos pequenos projectos. Das metodologias de suporte à gestão dos projectos e à própria cultura das organizações surgem problemas como:

- Âmbito do projecto mal definido
- Falta de acordo na definição de um conjunto bem articulado de objectivos do projecto
- Falta de dedicação às tarefas de gestão de projecto que na maioria das vezes são delegadas a elementos mais técnicos que não apresentam capacidades para tomar decisões críticas mais direccionadas para os gestores seniores.
- Gestores ignoram as boas práticas e as experiências passadas
- Fraca gestão, planeamento e controlo devido à falta de sistemas para medir o progresso e identificar potenciais riscos a tempo de reduzir os seus impactos.
- Fixação nas estimativas iniciais
- Alterações ao projecto são geridas de forma desatenta
- Estimativas irrealistas
- Nível de detalhe errado
- Falta de conhecimento acerca das ferramentas de gestão de projecto
- Demasiada confiança no software de gestão de projectos
- Demasiados recursos
- Elementos da equipa de projecto não apresentam as características técnicas necessárias
- Problemas entre os elementos da equipa
- Recompensa pelas acções erradas
- Requisitos mal definidos
- Gestores de projecto não entendem as necessidades dos clientes
- Problemas de qualidade descontrolados
- Adopção de novas tecnologias
- Tecnologia escolhida muda ou não é apropriada para suportar o projecto
- Necessidades de negócio mudam
- Utilizadores dos sistemas são cépticos e resistentes à mudança

As ferramentas de SW contribuem também para o insucesso referido na medida em que não apresentam no seu todo as seguintes características:

- Facilidade de Utilização
- Sem necessidade de treino / especialização
- Funcionalidades de planeamento e controlo
- Gestão de recursos
- Gestão de riscos
- Gestão de requisitos
- Espaço para documentação
- Suporte à comunicação

Como foi descrito no capítulo 3.2, as características que diferenciam os pequenos projectos de SI dos projectos de larga escala são:

- Orientados / sensíveis ao tempo,
- Oportunistas,
- Objectivo final bem definido,
- Nem todos os requisitos estão bem definidos no início,
- Podem ser extensões a produtos já existentes,
- Riscos e custos menos elevados do que nos grandes projectos,
- Controlo e planeamento flexíveis,
- Recursos e equipamentos partilhados por mais projectos,
- Fase de desenho bastante característica,
- Redes de comunicação fortes suportadas por mecanismos colaborativos,
- Possibilidade de substituição do gestor de projecto por um *juggler*.

De modo a alcançar o objectivo definido, foi proposta uma metodologia que segue as características identificadas nas necessidades dos pequenos projectos de SI e uma ferramenta de SW. Esta foi implementada segundo os requisitos levantados por análise dos pontos de falha de ferramentas comerciais aquando da sua aplicação a pequenos projectos. A ferramenta foi testada em ambiente empresarial durante um período de 2 meses onde foi também possível recolher testemunhos e novos requisitos por parte dos utilizadores.

As funcionalidades resultantes dos requisitos levantados foram implementadas e testadas com sucesso. A receptividade dos utilizadores foi positiva, quer do ponto de vista da facilidade de utilização, quer da interface. O *feedback* recebido por parte dos utilizadores foi fundamental, devido ao crescente interesse da parte deles em melhorar constantemente a aplicação.

Apesar de haver situações para as quais a ferramenta não deu um contributo claro, por exemplo na definição do âmbito ou requisitos irrealistas, o balanço final é bastante positivo pois ficou demonstrado que os problemas de comunicação podem ser minimizados com ferramentas colaborativas e de fácil utilização. O maior contributo do protótipo testado foi para

as actividades que envolvem decisões transversais à equipa de projecto e ao cliente e, mais concretamente, as tarefas respeitantes a estimativas temporais e de esforço.

7.1. Trabalho futuro

Como trabalhos futuros ficam algumas considerações que poderão trazer novos desenvolvimentos ao trabalho em causa:

- Elevar o número de casos de estudo de modo a reunir conhecimento de outras áreas de projecto generalizando-se assim o máximo possível a ferramenta implementada
- Implementar paralelamente a metodologia proposta no sentido de poder provar o seu valor científico enquanto complemento da disciplina de gestão de projectos
- Refinar modo de funcionamento da aplicação de forma a torná-la ainda mais intuitiva
- Melhorar a interface, por exemplo, acrescentar a representação gráfica das dependências entre tarefas no gráfico de Gantt

Para além destas propostas, existem outras um pouco mais complexas e interessantes, como por exemplo, para o problema de pedidos de alteração ao projecto, seria interessante adaptar o processo de *Change Management* do ITIL, testa-lo com um caso real e perceber quais as suas vantagens e desvantagens.

Referências

1. **Conti, Samantha.** *IS&T Project Management: Project Management 101*. Boston : MIT press, 2006.
2. *Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria.* **Atkinson, Roger.** s.l. : International Journal of Project Management, Elsevier Science, 1999, Vol. 17, pp. 337-342. 6.
3. *Managing and Modelling Project Risk Dynamics A System Dynamics-based Framework.* **Shenhar, Aaron J.** s.l. : Fourth European Project Management Conference, PMI Europe, 2001, Vol. 7.
4. *Dynamic modeling of product development processes.* **Ford, David N. e Sterman, John D.** s.l. : System Dynamics Review, 1998, Vol. 14, pp. 31-68. 1.
5. **Carvalho, Nuno Ponces de.** *"Gestão de Projectos Informáticos" course book*. Lisbon : s.n., 2005.
6. *A Process Framework for Small Projects, Software Process: Improvement and Practice.* **Leung, Hareton K. N. e Yuen, Terence C. F.** Hong Kong : The Hong Kong Polytechnic University, 2001, Vol. 6.
7. *A Software Process Improvement Approach Tailored for Small Organizations and Small Projects.* **Brodman, Judith G. and Johnson, Donna L.** 1997. Proceedings of the 1997 (19th) International Conference on Software Engineering.
8. **Pollice, Gary.** *Using the Rational Unified Process for Small Projects: Expand Upon extreme Programming.* s.l. : Rational Software Corporation, 2001.
9. *How to Fail with the Rational Unified Process: Seven Steps to Pain and Suffering.* **Larman, C., Kruchten, P. e Bittner, K.** s.l. : Valtech Technologies & Rational Software, 2001.
10. *Identifying Extensions Required by RUP (Rational Unified Process) to Comply with CMM (Capability Maturity Model) Levels 2 and 3.* **Manzoni, L. V. e Price, R. T.** s.l. : IEEE Transactions on Software Engineering, IEEE Press Piscataway, NJ, USA, 2003, Vol. 29, pp. 181-192. 2.
11. *A Comparison between Agile and Traditional Software Development Methodologies .* **Awad, M. A.** Australia : Honours Programme of the School of Computer Science and software Engineering, The University of Western Australia, 2005.

12. *Extreme Programming Explained: Embrace Change*. **Beck, Kent**. s.l. : Addison-Wesley, Reading, PA, 1999, Vol. 32, pp. 70-77. 10.

13. *Extreme Programming: A Humanistic Discipline of Software Development*. **Beck, Kent**. s.l. : Springer, 1998.

14. *The ToxicFarm Integrated Cooperation Framework for Virtual Teams*. **Godart, C., et al.** s.l. : Distributed and Parallel Databases, Springer, 2004, Vol. 15, pp. 67-88. 1.

15. *Caramba—A Process-Aware Collaboration System Supporting Ad hoc and Collaborative Processes in Virtual Teams*. **Dustdar, Schahram**. s.l. : Distributed and Parallel Databases, Springer, 2004, Vol. 15, pp. 45-66. 1.

16. *Virtual Teams: What are their Characteristics, and Impact on Team Performance?* **Wong, S. S. e Burton, R. M.** s.l. : Computational & Mathematical Organization Theory, Springer, 2000, Vol. 6, pp. 339-360. 4.

17. *An encompassing life cycle centric survey of software inspection*. **Laitenberger, O. e DeBaud, J. M.** s.l. : The Journal of Systems & Software, Elsevier, 2000, Vol. 50, pp. 5-31. 1.

18. *The Inspection Method Applied to Small Projects*. **Runge, B.** Los Alamitos, CA, USA : s.n., 1982. 6th international conference on Software engineering, IEEE Computer Society Press.

19. *Design and Code Inspections to Reduce Errors in Program Development*. **Fagan, Michael E.** s.l. : IBM Systems Journal, 1976, Vol. 15, pp. 182-211. 3.

20. **Westney, Richard**. *Computerized Management of Multiple Small Projects*. s.l. : CRC, 1992. Vol. 17.

21. **TopTenREVIEWS, Inc.** TopTenReviews. *Project Management Software Review*,. [Online] TopTenReviews. [Citação: 1 de January de 2007.] <http://project-management-software-review.toptenreviews.com/microsoft-project-review.html>.

22. **L.L.C., Virtual Communication Services**. MSP Professional vs VPMi©. *VCSonline.com*. [Online] Virtual Communication Services L.L.C. [Citação: 1 de January de 2007.] http://www.vcsonline.com/VCS/Reviews/Microsoft_Project_Server_2003.htm.

23. **Donovan, E. e Kaghan, W.N.** *Microsoft Project 2005: Eliminating Decreasing Returns Caused by Project Size and Complexity*. s.l. : Microsoft, 2005.

24. *Report on Microsoft Solutions Framework*. **Erharuyi, Edison, Vimjam, Siva RK Prasad and Wenjin, Yang**. Blekinge : s.n., 2004. Blekinge Institute of Technology.

25. **Lussier, Stephane.** Managing small projects I: Finding the juggler. *Critical Path: practical tips for software managers*. [Online] Macadamian, 29 de June de 2005. [Citação: 15 de November de 2006.] http://www.macadamian.com/index.php?option=com_criticalpath&task=view&id=20.
26. —. Managing small projects II: The juggling act. *Critical Path: practical tips for software managers*. [Online] Macadamian, 15 de June de 2005. [Citação: 11 de November de 2006.] http://www.macadamian.com/index.php?option=com_criticalpath&task=view&id=21.
27. **Mirski, Peter J.** *Strategy & Project Management: Project orientated organisations*. Finnland : MCI, University of Applied Sciences, 2005.
28. *Capability Maturity Model for Software*. **Paulk, M. C., et al.** s.l. : Software Engineering Institute, Carnegie Mellon University, 1991.
29. *Extreme programming from a CMM perspective*. **Paulk, N. C.** s.l. : Software, IEEE, 2001, Vol. 18, pp. 19-26. 6.
30. *Virtual Team Members: A Case Study in Management of Multiple Projects in a Limited Resource Environment*. **Briscoe, Thomas D.** 1997. PICMET Portland International.
31. *A Two-Person Inspection Method to Improve Programming Productivity*. **Bisant, D. B. e Lyle, J. R.** s.l. : Software Engineering, IEEE Transactions on, 1989, Vol. 15, pp. 1294-1304. 10.
32. *Quantitative Survey on Extreme Programming Projects*. **Rumpe, B. e Schroeder, A.** s.l. : Munich University of Technology, 2002.
33. **Kohrell, David e Wonch, Bill.** Using RUP to manage small projects and teams. *IBM - Developer Works*. [Online] IBM, 15 de July de 2005. [Citação: 12 de July de 2007.] <http://www.ibm.com/developerworks/rational/library/jul05/kohrell/index.html>.
34. *The rational unified process made easy: a practitioner's guide to the RUP*. **Kroll, P. e Kruchten, P.** s.l. : Addison-Wesley, 2003.
35. **Group, APMG.** What is PRINCE2. *PRINCE2 - PProjects IN Controlled Environments*. [Online] APMG Group, 23 de April de 2007. [Citação: 25 de May de 2007.] <http://www.prince2.org.uk/>.
36. *A Framework for Stakeholder Integration in Higher Education Information Systems Projects*. **Fowler, Alan e Gilfillan, Martin.** s.l. : Technology Analysis & Strategic Management, Taylor & Francis, 2003, Vol. 15, pp. 467-489. 4.

37. *A Software Development Process for Small Projects*. **Russ, Melissa L. e McGregor, John D.** s.l. : September / October IEEE Software, 2000, Vol. 17.
38. **Buehring, Simon.** Managing Small Projects. <http://www.projects-smart.co.uk>. [Online] 2003. [Citação: 15 de November de 2006.] <http://www.projects-smart.co.uk>.
39. *Organizing for managing multiple projects - A strategic perspective*. **Dietrich, P., Poskela, J. e Artto, K.A.** Reykjavik, Iceland : The 17th Conference on Business Studies, 2003, pp. 14-16.
40. *Successful management in multi-project environment*. **Dietrich, P., Karjalainen, J. e Artto, K.** Helsinki : Helsinki University of Technology, TAI Research Centre, 2002.
41. *Workflow Instance Scheduling with Project Management Tools, In: Database and Expert Systems Applications, 1998. Proceedings. Ninth International Workshop on*. **Bussler, Christoph.** 1998, pp. 753-758.
42. *The resource allocation syndrome: the prime challenge of multi-project management?* **Engwalla, Mats e Jerbrantb, Anna.** s.l. : International Journal of Project Management, 2003, Vol. 39.
43. **Schwaber, K. e Beedle, M.** *Agile Software Development with Scrum*. s.l. : Upper Saddle River, NJ, Prentice - Hall, 2001.
44. *The Scrum software development process for small teams*. **Rising, L. e Janoff, N. S.** 17, s.l. : IEEE Software, 2000, pp. 26-32.
45. **Schwaber, K. e Mar, K.** Experiences of using Scrum with XP. *SCRUM It's About Common Sense*. [Online] Control Chaos. [Citação: 31 de January de 2006.] <http://www.controlchaos.com/XPKane.htm>.
46. **Palmer, S. R. e Felsing, J. M.** *A Practical Guide to Feature-Driven Development*. s.l. : Upper Saddle River, NJ, Prentice-Hall, 2002.
47. **Stapleton, J.** *Dynamic systems development method – The method in practice*. s.l. : Addison Wesley, 1997.
48. **Fowler, Martin.** The New Methodology. *Martin Fowler.com*. [Online] ThoughtWorks, 13 de December de 2005. [Citação: 14 de March de 2006.] <http://www.martinfowler.com/articles/newMethodology.html>.
49. **Consortium, DSDM.** DSDM Tour. *Dynamic System Development Method Consortium*. [Online] DSDM Consortium 2007. [Citação: 20 de February de 2007.] <http://www.dsdm.org/>.

50. **Highsmith, J. A.** *Adaptive Software Development: A Collaborative Approach to Managing Complex Systems*. s.l. : Addison Wesley, 2000.
51. *Exploiting information systems and technology through business process improvement*. **Weerakkody, Vishanth e Hinton, C. Matthew**. s.l. : Knowledge and Process Management, 1999, Vol. 6, pp. 17-23. 1.
52. *Can project management be defined?* **R. P., Oisen**. s.l. : Project Management Quarterly, 1971, Vol. 2, pp. 12-14. 1.
53. *Time and budget: the twin imperatives of a project sponsor*. **Wright, J. N.** s.l. : International Journal of Project Management, 1997, Vol. 15, pp. 181-186. 3.
54. **Turner, J. R.** *The Hand book of Project-based Management*. s.l. : McGraw-Hill, 1993.
55. **P.W.G. Morris, G.H. Hough**. *The Anatomy of Major Projects*. New York : John Wiley, 1987.
56. *How can IS / IT projects be measured for success?* **Wateridge, J.** s.l. : International Journal of project Management, 1998, Vol. 16, pp. 59-63. 1.
57. *Measurement of project success*. **Wit, A. De**. s.l. : International Journal of Project Management, 1988, Vol. 6, pp. 164-170. 3.
58. *Measuring Success: Establishing and Maintaining A Baseline*. **McCoy, F. A.** Montreal, Canada : Project management Institute Seminar / Symposiun, 1987, pp. 47-52.
59. *Critical success factors across the project lifecycle*. **Pinto, J. K. e Slevin, D. P.** s.l. : Project Management Journal, 1988, Vol. 19, pp. 67-75. 3.
60. *System development methodolgy and project success: an assessment of situational approaches*. **Saarinen, T.** Amsterdam : Elsevier Science Publishers BV, In: Information and Management, 1990, Vol. 19, pp. 183-193. 3.
61. *The 3-D model of information systems success: the search for the dependent variable continues*. **Ballantine, J., et al.** PA, USA : Idea Group Publishing Hershey, In: Information Resources Management Journal, 1996, Vol. 9, pp. 5-14. 4.
62. *IT projects: sink or swim*. **Taylor, Andrew**. s.l. : Br Computer Soc, 2000. Vol. 42, p. 24.
63. *The Last Word on Project Management*. **Bounds, Gene**. s.l. : IIE Solutions, 1998.
64. *Critical issues in abandoned information systems development projects*. **Ewusi-Mensah, K.** New York : Communications of the ACM, ACM Press New York, NY, USA, 1997, Vol. 40, pp. 74-80. 9.

65. *Critical success factors in software projects*. **Reel S., John.** s.l. : Software, IEEE, 1999, Vol. 16. 3.
66. *Using large vs. small group projects in capstone and software engineering courses*. **Stein, Michael V.** Minneapolis : ACM, 2002.
67. **Lussier, Stephane.** *Managing Small Projects*. s.l. : Macadamian Technologies, 2005.
68. *Profiling the Competent Project Manager, In:Project Management Research at the Turn of the Millenium: Proceedings of PMI Research Conference*. **Crawford, Lynn.** Paris : Sylva, NC: Project Management Institute, 21-24 de June de 2000, Vols. 3-15, pp. 3-15.
69. *Tools of the trade: A survey of project management tools*. **Fox, Terry L.** 3, s.l. : Project Management Journal, 1998, Vol. 29.
70. *Management of multiple simultaneous projects: a state-of-the-art review*. **Payne, John H.** s.l. : Elsevier Science, International Journal of Project Management, 1995, Vol. 13.
71. **Mochal, Tom.** Manage small projects with a more informal service. *TechRepublic*. [Online] CNET Networks, Inc., 2 de August de 2005. [Citação: 15 de July de 2007.] http://articles.techrepublic.com.com/5100-10878_11-5810487.html.