FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Metodologia para Equipas de Desenvolvimento de Requisitos de Sistemas de Informação

Pedro Rodrigo Caetano Strecht Ribeiro

Relatório de Dissertação
Mestrado Integrado em Engenharia Informática e Computação

Orientador: João Carlos Pascoal de Faria (Professor Auxiliar)

Julho de 2008

Metodologia para Equipas de Desenvolvimento de Requisitos de Sistemas de Informação

Pedro Rodrigo Caetano Strecht Ribeiro

Relatório de Dissertação
Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Rosaldo José Fernandes Rossetti (Professor Auxiliar)

Arguente: Alberto Manuel Rodrigues da Silva (Professor Auxiliar)

Vogal: João Carlos Pascoal de Faria (Professor Auxiliar)

31 de Julho de 2008

Resumo

O trabalho apresentado nesta dissertação enquadra-se na área de Engenharia de Requisitos de produtos de software. Foi realizado no Projecto de Sistemas de Informação (PSI) da Faculdade de Engenharia da Universidade do Porto (FEUP).

A motivação para iniciar este trabalho foi a necessidade de definir o processo de especificação de requisitos para sistemas de informação desenvolvidos pelo PSI. Esse processo insere-se na etapa inicial do ciclo de desenvolvimento de software quando há a necessidade de descrever completamente um sistema antes de este ser desenhado, implementado e testado. Os requisitos de um sistema definem as suas restrições, os serviços que deve fornecer e informação de domínio de aplicação à equipa de desenvolvimento. Uma especificação de requisitos rigorosa contribui para que um sistema seja concebido de forma a responder adequadamente às necessidades dos seus clientes.

Os objectivos deste trabalho são a definição de uma metodologia de desenvolvimento de requisitos apropriada ao tipo de projectos desenvolvidos no PSI e a aplicação dessa metodologia na especificação de um sistema em particular.

Neste documento começa-se por descrever o processo e conceitos importantes de engenharia de requisitos de acordo com uma metodologia genérica, incluindo as técnicas recomendadas para cada actividade. Propõe-se, de seguida, uma metodologia que conjuga a introdução de aspectos inovadores com a reutilização de aspectos de outras metodologias já existentes. Esta abordagem impõe pressupostos ao tipo de sistema a ser especificado e tem como principal característica o enfoque no trabalho colaborativo realizado por uma equipa de especificação. Descrevem-se também detalhadamente as actividades e artefactos que constituem a metodologia proposta. Para a avaliar e validar os seus resultados, exemplifica-se um caso prático de aplicação na especificação de um sistema de informação a ser desenvolvido pelo PSI.

O principal resultado que se pretende alcançar com este trabalho é a definição de um processo de desenvolvimento de requisitos, que possa ser realizado de forma sistemática e com efeitos facilmente previsíveis, contribuindo dessa forma para o aumento da maturidade da organização. Como resultado adicional, pretende-se mostrar exemplos de artefactos produzidos durante a especificação de um sistema real para ilustrar a sua aplicação.

Abstract

The work presented in this thesis lies in the area of Requirements Engineering for software products. It was done in the Project of Information Systems (PSI) of the Faculty of Engineering of the University of Porto (FEUP).

The motivation to embark on this work was the need to define the requirements specification process for information systems developed by PSI. This process is carried out in the initial stage of the software development cycle when there is the need to completely describe a system before it is designed, implemented and tested. The requirements for a system set out the services that should be provided, define constraints and supply domain information to system developers. An accurate requirements specification allows a system to be conceived to adequately respond to the needs of its customers.

The goals of this work are the definition of a methodology for developing requirements considered appropriate to the type of projects developed by PSI and the application of this methodology in the specification of a particular system.

This document begins by describing the process and main concepts of requirements engineering according to a generic methodology, including recommended techniques for each activity. It is proposed then a methodology that combines the introduction of innovative aspects with the reuse of aspects of other existing methodologies. This approach requires assumptions to the type of system to be specified and its main feature is the emphasis on collaborative work done by a specification team. It also describes in detail the activities and artifacts that are part of methodology. To assess and validate the results, it was set up as case study the specification of an information system to be developed by PSI.

The main result to achieve with this work is the definition of a requirements development process to be performed systematically and with easily predictable outputs, thus contributing to increase the maturity of the organization. As further result, it is intended to show examples of artifacts produced during the specification of a real system to illustrate its application.

Agradecimentos

O autor gostaria de agradecer ao Prof. João Pascoal Faria a sua orientação pela disponibilidade, incentivo e contribuições inestimáveis.

Gostaria de agradecer ao Engº Manuel Machado e Mestre Tito Vieira pelas condições disponibilizadas para a realização deste trabalho.

Agradeço ainda ao Prof. Gabriel David, Engº Marco Nunes, Drª. Elisabete Neves, Engº Hugo Pereira, Drª. Carla Barbosa, Drª. Matilde Moreira e Dr. Bernardino Ribeiro todas as contribuições no contexto do processo de desenvolvimento de requisitos do sistema WebGA.

Um agradecimento especial aos meus amigos Prof. Paulo Ventura e Profª Rosário Aboim pela colaboração adicional na revisão dos textos desta dissertação e ao Engº Vítor Carvalho pelo apoio na revisão de elementos gráficos.

O Autor

Conteúdo

Lista de Figuras

Lista de Tabelas

Abreviaturas

BPM	*Business Process Management*
CASE	*Computer-Aided Software Engineering*
FEUP	Faculdade de Engenharia da Universidade do Porto
GAUP	Gestão de Alunos da Universidade do Porto
IBM	*International Business Machines*
IDE	*Integrated Development Environment*
IEEE	*Institute of Electrical and Electronics Engineers*
ISO	*International Organization for Standardization*
OCL	*Object Constraint Language*
PSI	Projecto de Sistemas de Informação
RUP	*Rational Unified Process*
SGBD	Sistema de Gestão de Bases de Dados
SIGARRA	Sistema de Informação para a Gestão Agregada de Recursos e Registos Académicos
SWEBOK	*Software Engineering Body Of Knowledge*
UML	*Unified Modeling Language*
UP	Universidade do Porto
W3C	*World Wide Web Consortium*
XML	*eXtensible Markup Language*

Capítulo 1

Introdução

1.1 Enquadramento

Este trabalho de dissertação enquadra-se na área de Engenharia de Requisitos de produtos de software. Foi realizado no Projecto de Sistemas de Informação (PSI) da Faculdade de Engenharia da Universidade do Porto (FEUP). O PSI tem como principal responsabilidade o desenvolvimento e manutenção do Sistema de Informação para a Gestão Agregada de Recursos e Registos Académicos (SIGARRA). O SIGARRA é um sistema de informação universitário baseado na *web*, com origens no Sistema de informação da FEUP (SiFEUP), que data de 1996. Suporta uma diversidade de processos, com particular predominância no processo pedagógico, integrando outros sistemas. É utilizado em todas as unidades orgânicas na Universidade do Porto e em outras instituições de ensino superior.

O PSI integra uma equipa de cerca de vinte colaboradores com diferentes responsabilidades nas áreas de gestão e coordenação de projectos, especificação de requisitos, análise, desenho, desenvolvimento e manutenção de sistemas de informação. Para além do SIGARRA, o PSI é responsável por sistemas em outros organismos da Universidade do Porto e noutras organizações públicas e privadas.

O autor exerce actividade no PSI desde 2000 e tem sido responsável pelo desenvolvimento de alguns módulos do SIGARRA, dos quais se destacam os de inscrições em disciplinas e lançamento de classificações. Tem também participado no desenvolvimento de outros sistemas de informação, o que lhe possibilitou tomar contacto com diferentes áreas de negócio e diversificar as tecnologias utilizadas. Recentemente tem assumido responsabilidades no que diz respeito à formalização de procedimentos das diferentes etapas do ciclo de desenvolvimento de software. A proposta desta dissertação alinha-se com essas responsabilidades e define, sob a forma de uma metodologia, procedimentos a serem adoptados no PSI para realização da etapa de especificação de requisitos para este tipo de sistemas.

1.2 Motivação e objectivos

Uma metodologia, em qualquer domínio de engenharia, destina-se a orientar a forma de realizar um processo, sob a forma de um conjunto de actividades, podendo sugerir uma ordem de execução e indicar os resultados esperados no fim. Concretamente, uma metodologia de desenvolvimento de requisitos define formalmente um processo que se destina a encontrar os requisitos de um sistema, analisá-los e descrevê-los pormenorizadamente antes de este ser construído. Como em outros domínios da engenharia, existe uma grande diversidade de metodologias, cada uma com as suas particularidades. A selecção de uma metodologia em detrimento de outras pode ser motivada por diversas condicionantes. No contexto da especificação de requisitos de sistemas de informação, algumas metodologias podem ser adequadas para grandes sistemas enquanto outras só o serão para pequenos módulos. Outro factor de decisão pode estar relacionado com a dimensão da equipa envolvida no projecto.

A aplicação efectiva de uma metodologia implica, frequentemente, que sejam efectuados ajustes na sua implementação. Esses ajustes são impostos pelas características específicas de cada projecto. Alguns projectos podem exigir que algumas actividades da metodologia sejam mais alargadas, enquanto outros deliberadamente poderão negligenciar partes. Por exemplo, um sistema que se destine a substituir outro já existente implica que este seja analisado com detalhe. Em sistemas para os quais não existam predecessores, essa análise poderá ser substituída por uma análise de sistemas similares no mercado. É quase sempre impraticável e pouco interessante tentar aplicar uma metodologia tal e qual como vem descrita, pois seria muito limitativo não poder acomodar alterações. Contudo existem metodologias que acomodam alterações mais facilmente que outras. Existem metodologias que cobrem todo o processo do ciclo de desenvolvimento de um sistema de software, como a *Rational Unified Process* (RUP) [1]. Outras são especificamente destinadas ao processo de engenharia de requisitos, como a de Kotonya-Sommerville [2]. Estas metodologias elencam as principais etapas que devem ser realizadas e sugerem actividades no contexto de cada uma. Qualquer organização que desenvolva projectos de software (*software houses*, centros de informática, etc.) pode tentar aplicá-la da forma como é descrita ou utilizá-la como ponto de partida para desenvolver a sua própria metodologia. Se a mesma organização desenvolver projectos com características muito diferentes, poderá desenvolver metodologias mais dirigidas a cada grupo de projectos. Ao desenvolver a sua própria metodologia, é provável que a necessidade de efectuar desvios ao previsto diminua. Dessa forma promove-se a formalização de processos contribuindo para o aumento da maturidade da organização. O processo de

desenvolvimento de requisitos passa a ser realizado de forma mais sistemática e com resultados mais facilmente previsíveis.

Os objectivos deste trabalho estão directamente relacionados com a sua motivação. Em Dezembro de 2007 foi proposto, ao Projecto de Sistemas de Informação, o desenvolvimento de um novo sistema de gestão de alunos (WebGA) que substitua o sistema actual (GAUP). Como qualquer novo projecto, identificou-se de imediato a necessidade de produzir um documento de especificação de requisitos para o novo sistema. O PSI já definiu algumas partes do processo de gestão de requisitos, suportando-o na sua ferramenta de gestão de projectos desenvolvida internamente. O processo de desenvolvimento de requisitos, contudo, ainda não se encontra definido de uma forma satisfatória e sistemática. Esta necessidade, alinhada com a perspectiva de melhoria contínua dos processos de desenvolvimento de software, levou a que o PSI reconhecesse que o novo sistema de gestão de alunos seria uma oportunidade para contextualizar e validar a proposta de uma nova metodologia de desenvolvimento de requisitos.

Este trabalho pretende dar resposta a essa necessidade, propondo uma metodologia para equipas de desenvolvimento de requisitos de sistemas de informação a ser adoptada no PSI. A motivação para derivar uma metodologia é, a partir do estudo de outras metodologias mais genéricas, encontrar uma que se adeque à dinâmica de trabalho desta organização e às características dos seus projectos. As actividades realizadas no PSI têm vindo progressivamente a configurar uma perspectiva colaborativa, em que as tarefas são realizadas sempre que possível envolvendo mais do que uma pessoa. Tal implica uma dinâmica de trabalho em que ocorram frequentemente reuniões para trocas de ideias e sugestões. Por seu turno, os projectos desenvolvidos no PSI apresentam características similares. Embora suportando diferentes áreas de negócio (por exemplo: processo pedagógico, bolsas de estudo, recursos humanos, gestão de contra-ordenações, gestão de sócios, etc.), têm consistido sempre do desenvolvimento de sistemas de informação constituídos por vários módulos e suportando um manancial de regras de negócio complexas e intrincadas. A metodologia proposta não é, por isso, uma metodologia genérica. Assenta em pressupostos que condicionam o tipo de sistemas aos quais pode ser aplicada e impõe a constituição de uma equipa de especificação. A metodologia está dirigida especificamente para o desenvolvimento de sistemas de informação de dimensão média que suportem processos com uma forte componente administrativa e com restrições impostas por legislação. Estes pressupostos alinham-se com a dinâmica de trabalho e características dos projectos do PSI.

Embora motivada pela necessidade de produzir a especificação de requisitos para o projecto WebGA, pretende-se que a metodologia seja apresentada de forma a poder ser aplicada em outros projectos de sistemas de informação, ou seja, abstraindo as especificidades desse projecto em particular.

Em resumo, os objectivos deste trabalho são:

1) Definição de uma metodologia de desenvolvimento de requisitos apropriada ao tipo de projectos desenvolvidos no PSI, tirando partido o mais possível de metodologias já existentes;
2) Aplicação dessa metodologia à especificação do sistema WebGA, permitindo dessa forma a validação e avaliação dos resultados alcançados.

Atendendo ao período para a sua realização e à dimensão do sistema WebGA não é pertinente apresentar uma versão completa do documento de especificação de requisitos. Importa, porém, que a versão apresentada inclua as partes essenciais deste documento e exemplifique os conteúdos em cada uma.

1.3 Estrutura

No capítulo 2 apresenta-se uma panorâmica de processos de engenharia de requisitos, com particular ênfase na metodologia de Kotonya-Sommerville. São descritos e analisados os sub-processos de desenvolvimento e de gestão de requisitos.

No capítulo 3 apresenta-se a metodologia de desenvolvimento de requisitos proposta pelo autor. São apresentados os pressupostos da sua aplicação e detalhadas as diferentes etapas e actividades em que se estrutura. Para cada actividade é descrito o formato estabelecido para cada artefacto que dela resulta.

No capítulo 4 descreve-se a aplicação concreta da metodologia proposta à especificação do sistema WebGA. Explica-se e ilustra-se a forma de aplicação da metodologia e simultaneamente permite apresentar os resultados obtidos na prossecução dos seus objectivos.

No capítulo 5 extraem-se as conclusões do trabalho efectuado e identificam-se tópicos para o trabalho futuro.

Os restantes modelos e metodologias utilizadas neste trabalho são descritas em anexo.

No anexo A é apresentada a metodologia de modelação de negócio de Eriksson-Penker utilizada para produzir modelos de negócio.

No anexo B é apresentada a metodologia de modelação de software com casos de uso, para enquadramento dos requisitos.

No anexo C é apresentada uma análise de modelos de documentos de especificação de requisitos para derivar um modelo na metodologia de desenvolvimento de requisitos proposta.

Capítulo 2

O processo de engenharia de requisitos

2.1 Introdução

O processo de desenvolvimento de software é constituído por um conjunto de actividades que têm por objectivo criar um produto de software. As várias formas de estruturar este processo são objecto de estudo da Engenharia de Software. Desse estudo têm vindo a ser propostos vários modelos que o possam transformar num processo sistemático e estruturado. Reconhecendo que as actividades devem ser realizadas de acordo com uma ordem pré-definida, começou a encarar-se o processo como um ciclo – o ciclo de desenvolvimento de software.

Um dos modelos para descrever o ciclo de desenvolvimento de um sistema de software é o modelo em cascata, que inclui várias etapas, numa sequência lógica do tipo "linha de montagem". Cada etapa tem objectivos próprios e subdivide-se em actividades, dependendo dos resultados da etapa anterior para poder iniciar-se. A figura 1 apresenta uma variante do modelo em cascata, onde é possível verificar que cada etapa tem entradas e saídas bem definidas que são, por sua vez, passadas à etapa seguinte.

No âmbito do desenvolvimento de um sistema de software, o ciclo de desenvolvimento inicia-se com a etapa de especificação de requisitos. As entradas desta etapa são fontes de requisitos e fontes de mudança, constituindo toda a informação considerada essencial para o desenvolvimento e manutenção do sistema. O seu objectivo é identificar as suas necessidades e requisitos globais de informação. O resultado desta etapa é um documento de especificação de requisitos que descreve em detalhe tudo o que o sistema deve fazer.

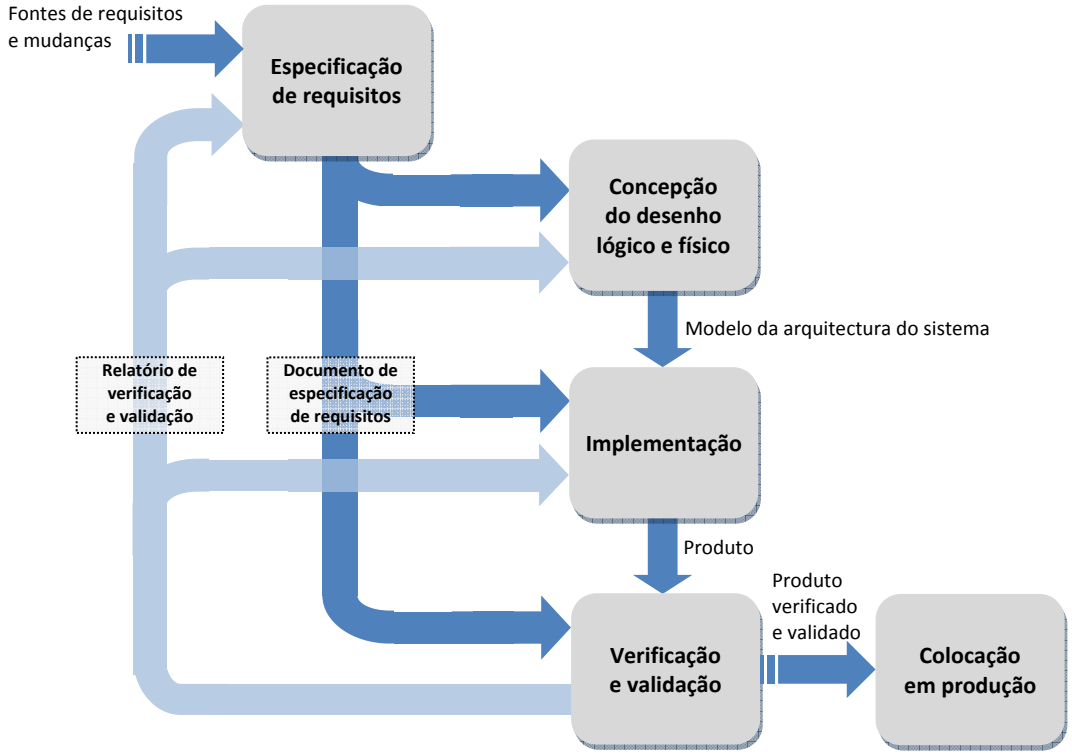

Figura 1 – Modelo do ciclo de desenvolvimento de software

Na etapa de concepção do desenho lógico e físico utiliza-se como entrada o documento de especificação de requisitos e desenham-se soluções que possam ser implementadas para criar um sistema de software. Envolve actividades de desenho lógico e físico. O desenho lógico incide na criação de modelos das arquitecturas de software e estruturas de dados de forma independente da tecnologia a ser utilizada. O desenho físico é específico das tecnologias que serão utilizadas e incide, entre outros aspectos, no detalhe metodologias para as interfaces, concepção de algoritmos e estratégias de armazenamento. O resultado desta etapa é a definição do modelo da arquitectura do sistema de software a ser implementado.

A etapa de implementação utiliza o documento de especificação de requisitos e o modelo da arquitectura do sistema para construir o sistema pretendido. Envolve actividades de codificação, desenho de interfaces e construção do modelo de base de dados em ambientes apropriados para desenvolvimento. O resultado desta etapa é um sistema de software que inclui o código desenvolvido e a documentação relacionada com a implementação e procedimentos de instalação.

A etapa de verificação e validação recebe o sistema de software e procura garantir que este possa ser colocado em produção. A verificação é um processo de avaliação de um sistema de software ou componente, no âmbito de uma fase do seu desenvolvimento, para determinar se os artefactos produzidos satisfazem as condições impostas no início dessa fase. Consiste na realização de actividades de inspecção e revisão de produtos de software. A validação é um processo de avaliação do produto para determinar o cumprimento dos requisitos descritos no

documento de especificação de requisitos. Envolve a realização de testes que exercitem o código desenvolvido na implementação. Esses testes pretendem revelar defeitos no software para que possam ser corrigidos de forma a garantir que esse software atinja o nível de qualidade pré-estabelecido. O resultado desta etapa é um relatório de verificação e validação que identifique os problemas detectados no produto e a partir do qual é possível decidir se o mesmo pode ser colocado em produção.

A etapa de colocação em produção corresponde à instalação do sistema de software no ambiente operacional para o qual foi construído. Considera-se que esta etapa marca o fim do ciclo de desenvolvimento e inicia o ciclo de manutenção do produto. Este ciclo engloba, entre outras, as actividades de suporte e manutenção. Quando é necessário corrigir defeitos ou introduzir novas funcionalidade no sistema em produção, inicia-se um novo ciclo de desenvolvimento especificamente destinado a esse efeito.

A realidade mostra que nenhum ciclo de desenvolvimento adere de forma ideal ao modelo em cascata. A sequência nem sempre pode ser cumprida, já que frequentemente há partes do sistema que podem iniciar o seu desenho enquanto outras ainda permanecem em especificação. Seria impraticável aguardar que cada etapa terminasse completamente antes de se dar início à seguinte. A realização destas actividades revela que nenhuma tem uma conclusão claramente definida. É sempre possível refinar e melhorar o resultado de cada uma. Durante o próprio desenvolvimento do sistema, vão ocorrendo inevitáveis alterações ao nível do que dele se pretende. Para melhor representar esta realidade foram derivados outros modelos que evidenciam o carácter iterativo do ciclo de desenvolvimento de software como, por exemplo, o modelo em espiral. Reconhecem que é incontornável revisitar algumas vezes todas as etapas do ciclo até se obter um sistema pronto a ser entregue ao cliente. O modelo representado na figura 1 não é o modelo em cascata estrito pois já prevê a entrada do relatório de verificação e validação nas etapas da iteração seguinte.

Independentemente do modelo utilizado, verifica-se que em qualquer projecto desta natureza, a etapa de especificação de requisitos é sempre a primeira do ciclo de desenvolvimento de software e que as decisões nela tomadas condicionam o desenrolar das etapas seguintes.

2.2 A importância dos requisitos no ciclo de desenvolvimento de software

Os requisitos de um sistema são toda a informação que descreve completamente o comportamento que dele é esperado antes de ser desenhado, implementado e testado. A informação contida nos requisitos é muito variada podendo incluir descrições sobre o domínio de aplicação, propriedades, atributos e funcionalidades

do sistema. Pode incluir também restrições ao seu modelo de desenvolvimento e à forma de o operacionalizar.

Seguem-se alguns aspectos de um sistema aos quais estão associados requisitos:

- A informação do domínio de aplicação corresponde à descrição de regras de negócio, legislação e documentação oficial da organização onde conste a terminologia específica da área de negócio.
- As propriedades de um sistema são descrições muito genéricas de características que este deve possuir. Incluem, por exemplo, aspectos ligados ao desempenho do sistema, disponibilidade, tempo de resposta, fiabilidade, segurança, controlo de acessos e definição de privilégios. Podem também traduzir-se na obrigatoriedade de cumprimento de normas internacionais ou outros padrões. Os atributos do sistema correspondem às características pretendidas pelo cliente e permitem distinguir um produto de outro similar. Incluem, por exemplo, considerações ligadas à portabilidade, facilidade de manutenção e níveis de fiabilidade.
- As funcionalidades de um sistema representam os objectivos que os utilizadores pretendem com a sua utilização. Além do apoio a processos de negócio, incluem, por exemplo, comunicação com outros sistemas ou com outros dispositivos de hardware.
- As restrições ao desenvolvimento descrevem as tecnologias ou plataformas em que o sistema terá que ser desenvolvido. Em sistemas de software, tais requisitos incidem habitualmente na indicação explícita da linguagem de programação a utilizar na implementação do sistema, da tecnologia do sistema de gestão de base de dados e do sistema operativo.
- As restrições operacionais descrevem as condições em que o sistema será colocado em produção e que podem influenciar o seu nível de desempenho. As condições podem incidir nos limites dos recursos de hardware disponíveis, por exemplo, memória RAM, espaço disponível em disco ou capacidade de processamento. Em alguns sistemas poderá ser necessário indicar os intervalos de valores entre os quais determinadas grandezas físicas do ambiente de produção, como a temperatura e a humidade, podem variar.

Para auxiliar a distinguir o seu âmbito e importância é frequente classificar os requisitos de acordo com dois critérios:

- Classificação por tipo – Organiza os requisitos em funcionais e não-funcionais. Os requisitos funcionais dizem respeito às descrições do que o sistema deve fazer. Incidem por isso em tudo aquilo que é facilmente percepcionado pelo cliente, ou seja, as funcionalidades que o sistema oferece. Os requisitos não-funcionais dizem respeito à forma de implementar os requisitos funcionais. Embora possam ser formulados pelo cliente,

normalmente são de maior interesse para a equipa de desenvolvimento do sistema, como as imposições sobre a tecnologia a utilizar. Verifica-se, na prática, que nem sempre é simples classificar um requisito como funcional ou não-funcional.

- Classificação por nível de abstracção – Organiza os requisitos de acordo com o detalhe das suas descrições. Não havendo definida um escala quantificável podem, contudo, identificar-se os dois extremos desta classificação. Os requisitos de alto nível pretendem comunicar descrições muito genéricas. Incluem a missão e principais objectivos do sistema com enumerações e delineações com elevado grau de generalidade. À medida que se vai descendo de nível, os requisitos vão sendo progressivamente refinados aumentando o nível de detalhe nas suas descrições. Os requisitos de baixo nível são explicações muito detalhadas do comportamento do sistema.

A figura 2 representa cada critério de classificação em eixos ortogonais onde se procura sugerir de forma genérica a posição típica de alguns exemplos de descrições. Em alguns casos é claramente difícil situar apenas num dos quadrantes.

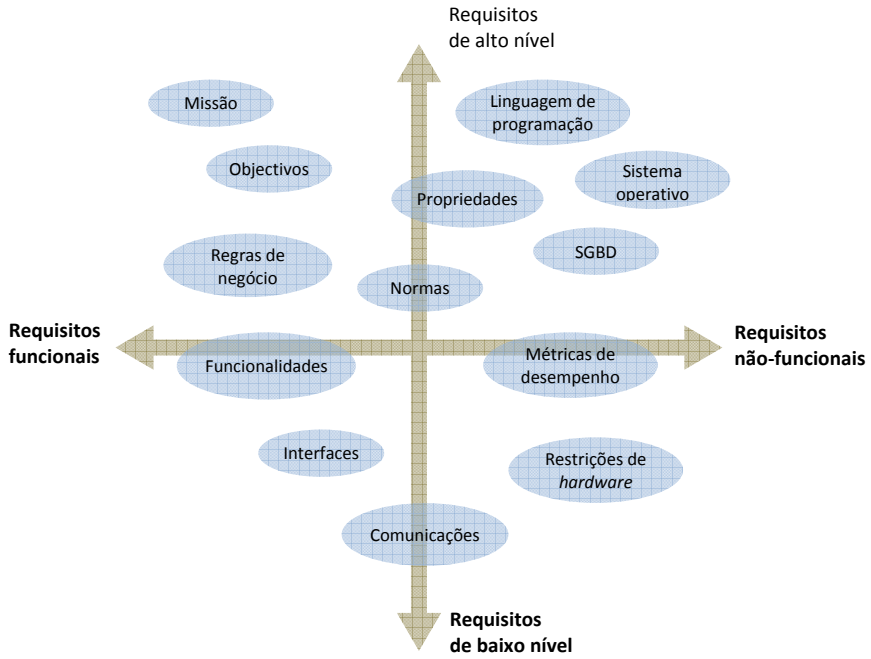

Figura 2 – Critérios de classificação de requisitos

Os requisitos são descritos e organizados no documento de especificação de requisitos. Este documento é de importância fundamental durante todo o ciclo de desenvolvimento do sistema de software. Contudo, mesmo fora desse contexto, a

produção deste documento é imprescindível para as organizações envolvidas no sistema por vários motivos [3]:

- Permite estabelecer uma base de acordo – O documento de especificação é utilizado como base contratual entre os fornecedores de software e os seus clientes. Ao comprometer ambas as partes, formaliza um acordo sobre a percepção do sistema a ser desenvolvido.
- Fornece documentação ao projecto – O documento de especificação agrega parte da informação necessária para a execução das actividades das restantes etapas do ciclo de desenvolvimento de software. Facilita, dessa forma, a transmissão do conhecimento existente sobre o sistema.
- Promove a compreensão do negócio – A elaboração do documento de especificação implica uma análise e compreensão do negócio, delimitando também as fronteiras do sistema de forma clara e precisa. Permite que o conhecimento sobre o sistema se mantenha no fornecedor de software de forma independente das pessoas envolvidas na concepção do mesmo.
- Apoia a gestão de projectos – O documento de especificação oferece uma descrição realista do produto a desenvolver constituindo uma importante ferramenta de apoio à estimação de custos, afectação de recursos e calendarização das etapas seguintes do ciclo de desenvolvimento.

Para que possa ser útil e servir os seus propósitos, o documento de especificação de requisitos deve possuir um elevado nível de qualidade. Para que tal seja possível, é necessário que a identificação de requisitos seja efectuada de forma metódica e estruturada. Segundo Kotonya e Sommerville [2] cerca de 25% do tempo de duração do projecto deve ser atribuído a esta etapa.

A compreensão incompleta, ou mesmo incorrecta do sistema por parte do fabricante de software pode trazer graves consequências ao projecto e levar ao fracasso do sistema. Ao subsistirem erros na compreensão do que o cliente pretende, a equipa de desenvolvimento poderá implementar requisitos que não reflectem as suas reais necessidades. Poderão permanecer requisitos inconsistentes ou incompletos. Tal pode conduzir à persistência de ambiguidades nos requisitos levando a que seja a equipa de desenvolvimento a tomar decisões que caberiam ao cliente. A necessidade de efectuar alterações não previstas, resultantes de especificações com pouca qualidade, contribui para o incumprimento de prazos e aumento dos custos de desenvolvimento. Se a equipa de desenvolvimento e testes for muito pressionada para cumprir prazos ou orçamentos, haverá o risco de o sistema não ser fiável e apresentar frequentemente erros ou "crashes" que interfiram com a operacionalização. O cliente e os utilizadores finais não ficarão satisfeitos com o sistema produzido, colocando em risco a sua utilização e credibilidade. Um

sistema nestas condições, ao continuar em produção, apresenta custos de manutenção muito elevados e com perspectivas de evolução limitadas. Do estudo apresentado por Patton [4], verifica-se que mais de 50% dos problemas detectados no software devem-se a erros na especificação, como se representa na figura 3. Os restantes são injectados em outras etapas do ciclo de desenvolvimento de software.

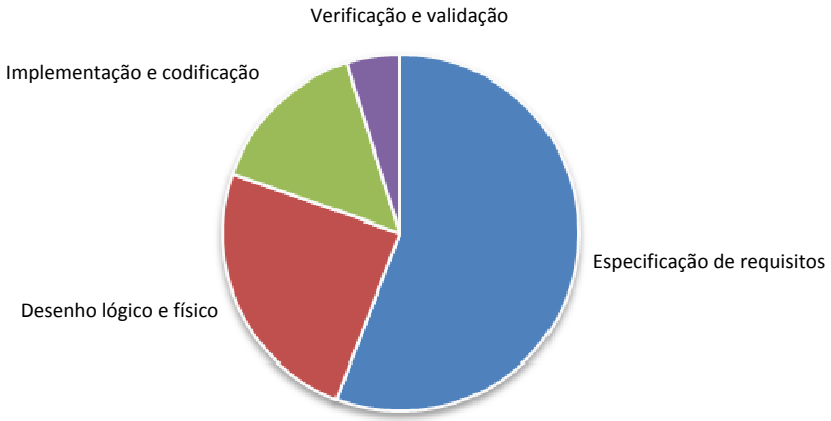

Figura 3 – Origem dos problemas nas etapas de desenvolvimento

Outro estudo, realizado por McConnell [5], tenta prever o custo da correcção de erros relacionando-os com as etapas em que são criados e posteriormente detectados. A conclusão do seu estudo é que esse custo aumenta significativamente à medida que se progride no ciclo de vida de desenvolvimento de software. O gráfico da figura 4 relaciona as etapas em que os erros são criados com as etapas em que são detectados, evidenciando a variação de grandeza no custo da correcção desses erros.

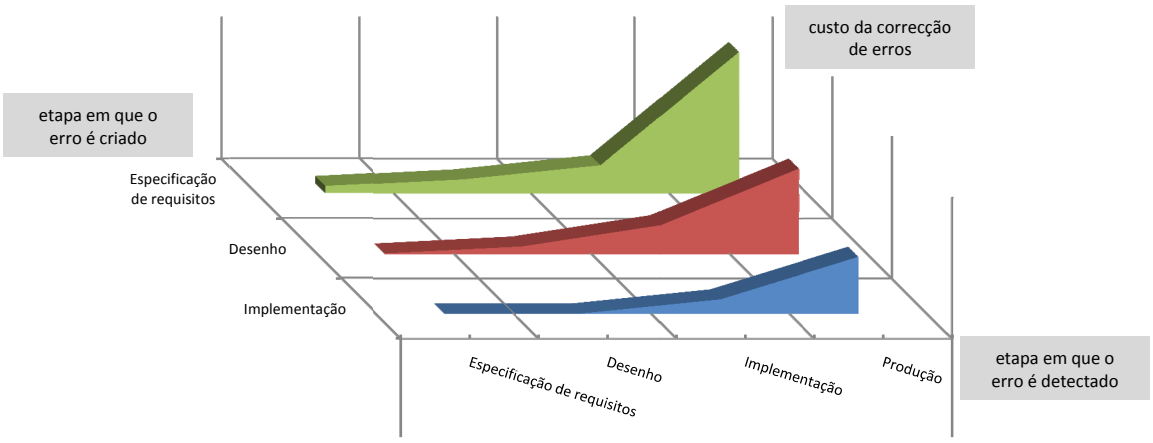

Figura 4 – Correcção de erros nas etapas do ciclo de desenvolvimento

Os erros criados durante a etapa de especificação de requisitos e detectados ainda nessa fase têm custo de correcção quase nulo. De facto, basta alterar a descrição de um ou mais requisitos para efectuar a correcção. Os erros detectados já na fase de desenho têm um custo mais elevado porque podem levar à reorganização da arquitectura concebida. Os erros de especificação detectados durante a codificação são ainda mais onerosos porque, além de eventuais alterações à arquitectura, obrigam a alterações de código frequentemente em mais de um módulo do sistema. Os erros de especificação detectados quando o sistema já está em produção são os que implicam maiores custos de correcção e os mais nefastos para o sistema. Corrigir problemas em produção, além de obrigar a rever a arquitectura e fazer alterações de código, pode implicar a paragem do sistema para se instalar as alterações decorrentes das correcções. Há que incluir também o custo dos efeitos nefastos provocados pelos erros no sistema e pelo período em que teve que ser parado.

Se por um lado se conclui que é na fase de especificação de requisitos que é introduzida a maior parte dos erros no sistema, por outro, quanto mais tarde estes forem detectados, mais onerosa se torna a sua correcção. Esta realidade justifica que a etapa de especificação de requisitos consuma uma importante parte do tempo do ciclo de desenvolvimento de software. À medida que progridem no nível de maturidade, os fabricantes de software investem cada vez mais na produção de especificações de requisitos de qualidade e encaram-nas como factor crítico de sucesso para os seus projectos.

2.3 O processo de engenharia de requisitos

O processo de engenharia de requisitos consiste numa abordagem estruturada às actividades de captura, organização, documentação e manutenção dos requisitos de um sistema. A utilização de técnicas e modelos permite prever e sistematizar a execução dessas actividades. Wiegers [6] defende que o processo de engenharia de requisitos deve ser separado em dois sub-processos:

- Desenvolvimento de requisitos, que se destina a criar novos requisitos no início do ciclo de desenvolvimento de software e em iterações seguintes;
- Gestão de requisitos, que se destina a controlar as alterações ao conjunto de requisitos criados previamente e avaliar a pertinência da sua incorporação no sistema.

À semelhança do próprio ciclo de desenvolvimento de software, cada um dos sub-processos é também constituído por actividades. A figura 5 apresenta a decomposição do processo de engenharia de requisitos nestes sub-processos e respectivas actividades.

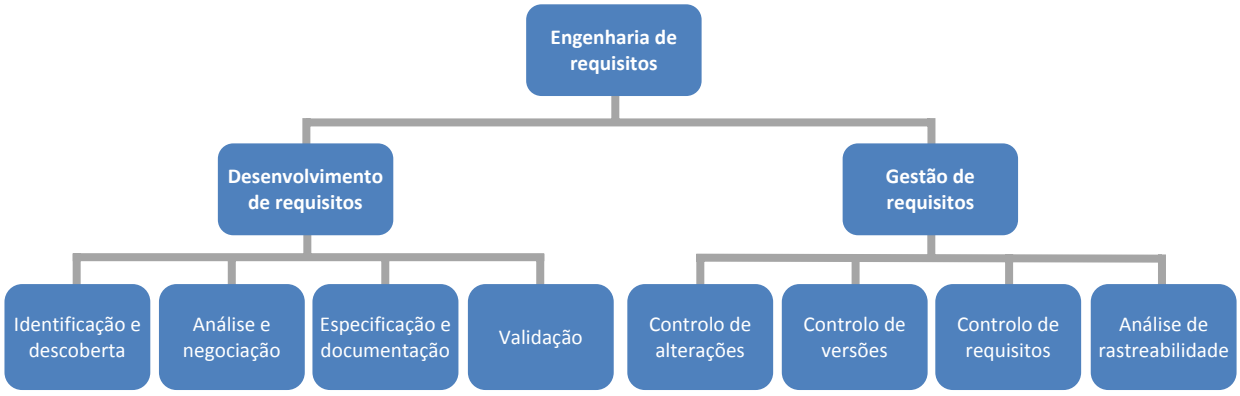

Figura 5 – Decomposição do processo de engenharia de requisitos

O processo de engenharia de requisitos pode ser utilizado para realizar a etapa de especificação de requisitos do ciclo de desenvolvimento de software, como representado na figura 6. Na primeira iteração deste processo, o sub-processo de desenvolvimento utiliza como entrada as fontes de requisitos para fornecer a primeira versão do documento de especificação de requisitos.

Figura 6 – Enquadramento do processo de engenharia de requisitos

Em iterações seguintes, o sub-processo de gestão de requisitos utiliza como entrada as fontes de mudanças para analisar alterações aos requisitos. As alterações serão incluídas em versões seguintes do documento de especificação de requisitos. Se existirem novos requisitos, são desenvolvidos e integrados também em versões seguintes. Da conjugação das actividades realizadas em ambos os sub-processos surge uma nova versão do documento de especificação de requisitos. Cada versão representa uma imagem do sistema na data em que o documento é produzido.

2.4 O processo de desenvolvimento de requisitos

O processo de desenvolvimento de requisitos é o processo que serve de base a todo o ciclo de desenvolvimento de software. A proliferação de metodologias de processos de engenharia de requisitos implica que não exista um processo ideal que possa ser imposto a uma organização. Algumas utilizam processos muito pouco

estruturados, baseados quase unicamente na experiência das equipas envolvidas. O sucesso dos resultados do processo está inteiramente dependente dessa equipa e pode não ser repetido se esta for alterada. Outras organizações utilizam processos sistemáticos baseados na aplicação de metodologias de análise de sistemas. O sucesso dos resultados tende a ser mais independente das equipas envolvidas.

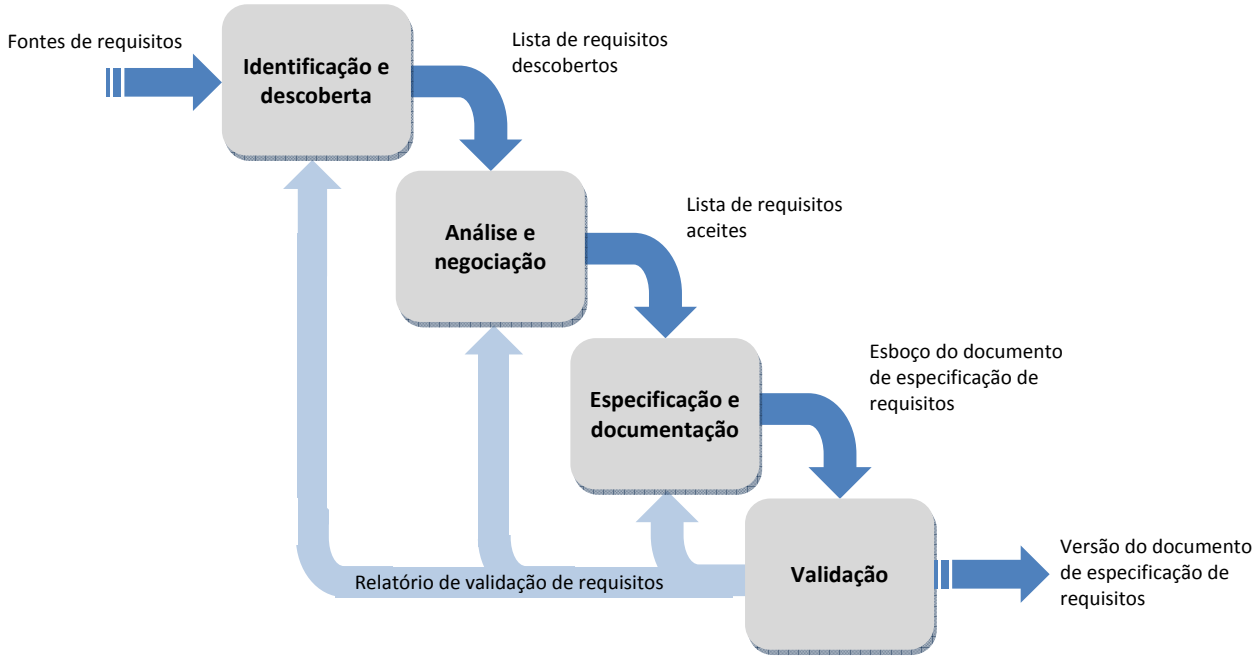

Figura 7 – Modelo em cascata do processo de desenvolvimento de requisitos

O processo genérico representado na figura 7 procura identificar e sequenciar as quatro actividades do processo de desenvolvimento de requisitos e é baseado na metodologia descrita por Kotonya e Sommerville [2]. Segundo os seus autores, as organizações devem considerá-lo como ponto de partida para a aplicação de forma apropriada às suas necessidades, não sendo necessário que uma actividade termine para que a seguinte possa ter início. Tal não implica, contudo, que à semelhança do modelo em cascata do ciclo de desenvolvimento de software, as saídas de uma actividade são as entradas da seguinte.

A sequência de actividades do processo de desenvolvimento de requisitos proposta por esta metodologia é a seguinte:

- Identificação e descoberta – A informação das fontes de requisitos é analisada e por aplicação de técnicas sistemáticas vão sendo descobertos os requisitos do sistema. É criada uma lista que identifica todos os requisitos e lhes associa uma descrição.
- Análise e negociação – A lista dos requisitos descobertos é analisada em detalhe para detecção de problemas. Os diferentes interessados no sistema

negociam os requisitos discutíveis para definir quais devem ser aceites e quais devem ser rejeitados. É criada uma lista dos requisitos aceites que é aprovada por todos os interessados.

- Especificação e documentação – A lista de requisitos aceites é organizada num documento. As descrições dos requisitos são formalizadas com o nível de detalhe adequado e de modo a serem compreensíveis por todos os interessados. O resultado é um esboço do documento de especificação de requisitos.
- Validação – O esboço do documento de especificação de requisitos é inspeccionado para verificar se tem um nível de qualidade que permita a sua utilização pelas restantes etapas do ciclo de desenvolvimento de software. Caso tal se verifique, é disponibilizada uma versão actualizada do documento de especificação de requisitos. Caso contrário, é elaborado um relatório de validação de requisitos que indique o que terá que ser revisto em cada actividade anterior.

Mesmo que o documento de especificação de requisitos já possua um nível de qualidade adequado, é difícil dar o processo da sua escrita por concluído. É sempre possível revisitar os requisitos de forma a encontrar formas de melhor os descrever e aumentar a sua qualidade. Na prática, verifica-se que, por restrições temporais, é necessário começar a fazer entregas de versões do documento de requisitos para poderem iniciar-se as fases seguintes. Embora o modelo da figura 7 seja um modelo em cascata, evidencia também o carácter repetitivo do processo pela re-alimentação do relatório de validação de requisitos. É possível, porém, evidenciar melhor as diferentes iterações representando o processo com um modelo em espiral por analogia com o equivalente para o ciclo de desenvolvimento de software [2], como o representado na figura 8.

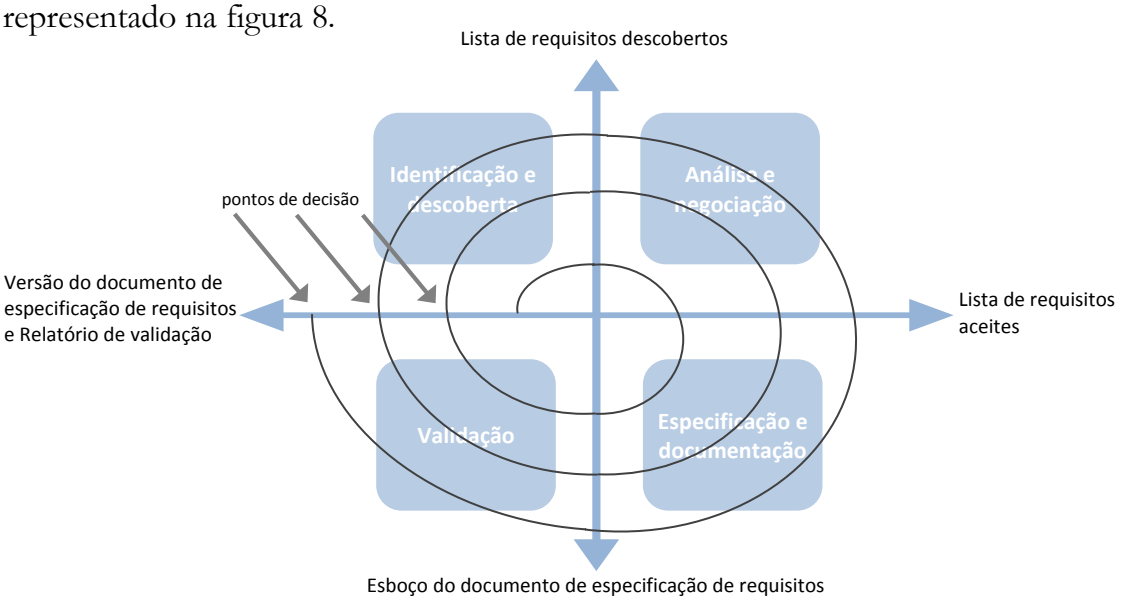

Figura 8 – Modelo em espiral do processo de desenvolvimento de requisitos

O modelo em espiral mostra que as diferentes actividades do processo se repetem até que seja tomada uma decisão que implique a aceitação de uma versão do documento de especificação de requisitos. Os pontos de decisão ocorrem no fim de cada ciclo completo da espiral em que se realizaram cada uma das quatro actividades. Ao verificar-se que o documento ainda não está em condições de ser entregue, o processo é reiniciado dando origem a nova iteração. A cada iteração aumenta o número de requisitos descobertos, de requisitos aceites e de requisitos correctamente documentados.

Idealmente, haverão tantas iterações quantas as necessárias até se obter um documento de especificação de requisitos que seja aceite. Na prática, as pressões dos prazos ou da falta de recursos forçam o término do processo. Quaisquer alterações posteriores serão integradas no processo de gestão de requisitos.

2.4.1 Identificação e descoberta de requisitos

A identificação e descoberta de requisitos constituem o primeiro conjunto de actividades do processo de desenvolvimento de requisitos de um sistema. É comum encontrar-se a expressão "levantamento de requisitos" para designar estas actividades. Contudo, tal sugere que se trata de uma actividade passiva de indicação dos mesmos. A prática mostra que os requisitos só existem a partir do momento em que são descobertos. De facto, as actividades associadas à identificação e descoberta de requisitos estão longe de ser passivas. Exigem grande dinamismo, empenho e rigor, não sendo consideradas de fácil execução. O engenheiro de requisitos é um profissional com competências específicas para as desempenhar. São funções que não devem ser realizadas por elementos da equipa de desenho do sistema, implementação ou testes, sob pena de a descrição dos requisitos do sistema incluir demasiados detalhes de desenho ou implementação.

A descoberta dos requisitos é um esforço conjunto do engenheiro de requisitos e dos interessados. A primeira tarefa do engenheiro de requisitos é descobrir os objectivos dos interessados na utilização do sistema a desenvolver. Deve atender-se a que este exercício não é simples porque nem sempre os interessados têm uma ideia clara de como o sistema os pode apoiar no seu trabalho. Diferentes interessados têm frequentemente perspectivas diferentes dos processos, que levam, muitas vezes à elaboração de requisitos conflituantes. O sucesso desta fase inicial do processo de desenvolvimento de requisitos é fundamental para o sucesso do sistema pois, se os requisitos reais do cliente não forem descobertos, é pouco provável que este fique satisfeito com o mesmo.

Segundo Kotonya e Sommerville [2], as actividades de descoberta de requisitos podem ser agrupadas de acordo com quatro "dimensões". A primeira envolve a compreensão do domínio de aplicação e consiste na aquisição de conhecimento básico sobre as áreas de negócio do sistema. Se o engenheiro de requisitos não

possuir esse conhecimento terá de efectuar um estudo preliminar para se familiarizar com os conceitos base. A segunda envolve a compreensão do enquadramento do sistema na organização e dos problemas que se propõe resolver. Incide nos detalhes específicos do sistema dentro do seu domínio de aplicação. É um estudo mais especializado que o anterior. A terceira envolve a compreensão do negócio e consiste em situar o sistema nos objectivos da organização e verificar a sua contribuição para os mesmos. A quarta envolve a compreensão das necessidades e restrições dos interessados e consiste em identificar quais são os processos que devem ser suportados pelo sistema para apoiar o trabalho dos interessados.

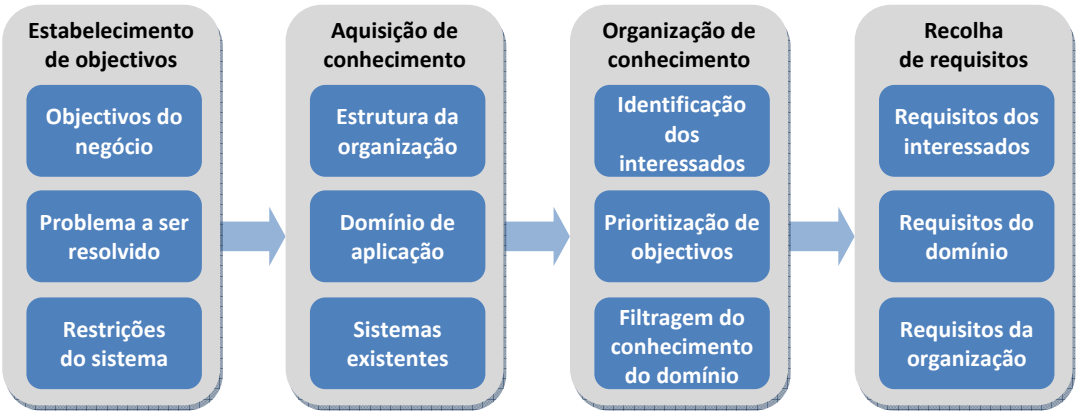

Figura 9 – Modelo das actividades de descoberta e identificação de requisitos

A informação necessária para dar resposta a estes problemas está espalhada por toda a organização. O modelo da figura 9 propõe uma abordagem estruturada às actividades de descoberta de requisitos. A abordagem sugerida por este modelo implica uma sequência de actividades:

- Estabelecimento de objectivos – Analisam-se os objectivos gerais das organizações da área de negócio que utilizará o sistema. Descreve-se, de forma genérica, o problema a ser resolvido e a contribuição que o sistema a desenvolver terá para a sua resolução. Identifica-se, entre outras, as restrições do sistema a nível orçamental e de prazo de execução.
- Aquisição de conhecimento – Obtém-se informação sobre a estrutura da organização, do domínio de aplicação da área de negócio e dos sistemas existentes, incluindo os que o novo sistema possa vir substituir.
- Organização do conhecimento – Organiza-se a informação obtida da actividade anterior, identificando cada interessado no sistema, qual o seu papel na organização e definindo-se prioridades dos objectivos. Retém-se a parte do conhecimento do domínio de aplicação que terá interesse como fonte de requisitos.

- Recolha de requisitos – Reúnem-se as fontes de requisitos e utilizam-se técnicas de recolha para derivar os requisitos dos interessados, do domínio de aplicação e da organização.

À semelhança de outros modelos, o representado na figura 9 é um modelo idealizado. Na prática, não é possível realizar as actividades na ordem indicada nem com tão elevado nível de estruturação. Verifica-se que a primeira actividade a realizar terá que ser a identificação das fontes de requisitos. O *Software Engineering Body of Knowledge* (SWEBOK)[1] [7] identifica as principais fontes de requisitos:

- Objectivos gerais do sistema – São os objectivos de alto nível do sistema e os factores críticos de sucesso. Os objectivos mostram a motivação para a implementação do sistema sob a forma de descrições genéricas.
- Interessados no sistema – São as pessoas ou organizações que serão afectadas pelo sistema e que têm uma influência directa ou indirecta nos seus requisitos. Incluem os utilizadores finais, gestores da organização, membros da equipa de desenvolvimento do sistema, clientes da organização que utilizarão o sistema para dele obter serviços e entidades externas como autoridades de certificação e regulação.
- Conhecimento do domínio de aplicação – Informação que é possível adquirir sobre o domínio de aplicação que não é transmitida pelos interessados no sistema. Permite avaliar compromissos que serão necessários para resolver conflitos entre os requisitos. Pode ser obtido por consulta de livros ou outras publicações e contacto com especialistas do domínio de aplicação.
- Ambiente operacional – Corresponde à informação que é possível obter nos locais onde o sistema será colocado em produção. Permite enquadrar o sistema a ser desenvolvido no contexto de outros sistemas com os quais poderá ter que comunicar.
- Ambiente organizacional – Conhecimento da cultura e políticas internas da organização que podem impor condições à forma como o sistema suporta os seus processos de negócio. O sistema deve estar alinhado com o ambiente organizacional de forma a não pressionar alterações não previstas aos processos de negócio.

Para além das referidas no SWEBOK poder-se-á ainda considerar três outras fontes:

- Ambiente externo – Conhecimento das restrições impostas por agentes externos à organização e das quais esta não tem controlo ou poder de

[1] O *Software Engineering Body of Knowledge* (SWEBOK) é um documento de referência da área de Engenharia de Software criado pelo *Institute of Electrical and Electronics Engineers* (IEEE).

decisão. Permite conhecer normas, regulamentos e legislação que se aplique aos processos de negócio da organização que utilizará o sistema.

- Produtos de mercado – Corresponde à informação que é possível extrair de produtos concorrentes ao sistema. Permite aferir como deve ser desenvolvido o sistema de forma a posicioná-lo e diferenciá-lo de outros no mercado.
- Sistemas internos – Corresponde à informação que é possível obter dos sistemas já existentes na organização, incluindo aqueles que poderão vir a ser substituídos pelo sistema a desenvolver.

Após identificar as fontes, o engenheiro de requisitos poderá iniciar a descoberta de requisitos. Segundo os autores do SWEBOK, esta é uma das áreas mais complexas do processo de desenvolvimento de requisitos. Mesmo quando o engenheiro de requisitos tem muitas fontes à sua disposição, é necessário que possua um espírito crítico para seleccionar a informação que é relevante para o sistema. Para apoiar o seu trabalho, podem ser utilizadas várias técnicas:

- Análise de documentação – Envolve as actividades de recolha, consulta e análise de informação da documentação que incida sobre o domínio da aplicação, sobre a organização que fará uso do sistema e sobre o ambiente externo, para dela extrair requisitos.
- Análise de sistemas existentes – Envolve actividades de interacção e consulta dos manuais dos sistemas existentes na organização e, quando possível, de outros sistemas da mesma área de aplicação, para compreender os motivos de estes satisfazerem alguns requisitos e não suportarem outros.
- Realização de entrevistas individuais – Envolve a realização de reuniões com cada interessado no sistema para obter a sua perspectiva, compreender as suas funções na organização e recolher as suas necessidades. O engenheiro de requisitos assume o papel de entrevistador e pode necessitar de uma preparação prévia para se familiarizar com a terminologia utilizada pelo seu interlocutor e para seleccionar o melhor formato para a entrevista. A entrevista pode consistir num questionário com perguntas fechadas das quais se pretendam respostas assertivas ou ser aberta, concedendo mais liberdade ao intervenientes. O sucesso de uma entrevista pode depender de factores que estão para além do controlo do entrevistador como a motivação do entrevistado, tempo disponível e interesse em colaborar.
- Realização de reuniões conjuntas – Envolve a realização de reuniões com um grupo de interessados no sistema para refinar ideias que são mais difíceis de abordar em entrevistas individuais. O engenheiro de requisitos assume o papel de mediador, garantido que todos os interessados têm oportunidade de intervir e detectar quando existem conflitos nos requisitos.

- Descrição de cenários – Envolve a adopção de uma abordagem a utilizar nas entrevistas ou nas reuniões conjuntas que coloque os interessados perante uma situação realista em que simulam a sua interacção com o sistema. Consistem em narrativas que explicam como o sistema deve ser utilizado em cada situação. Descrevem o fluxo normal de eventos mas também os fluxos excepcionais. O engenheiro de requisitos pode formalizá-los com diagramas de casos de uso e diagramas de actividade da *Unified Modeling Language*[2] (UML) [8].
- Prototipagem – Envolve a produção de uma versão inicial e muito básica do sistema para clarificar requisitos obscuros e enquadrá-los num contexto. Caracteriza-se por ser de rápido desenvolvimento, com funcionalidades limitadas e não ter em conta os requisitos não-funcionais. A sua apresentação aos interessados auxilia o engenheiro de requisitos a validar a compreensão da informação recebida.
- Observação e análise social – Envolve a observação do ambiente operacional directamente nas instalações da organização permitindo capturar tarefas que dificilmente são verbalizadas pelos interessados devido ao carácter rotineiro que assumem no contexto das suas actividades. Permite conhecer melhor como funciona a organização e o trabalho que nela é realizado.
- Reutilização de requisitos – Envolve a recuperação de requisitos que foram identificados para um sistema e utilizá-los novamente em outro sistema. Os benefícios são a poupança de tempo e esforço pois os requisitos reutilizados já foram analisados e validados nos outros sistemas. Os requisitos mais passíveis de reutilização são os que estão associados ao conhecimento do domínio de aplicação, estilos de apresentação de informação e os que reflectem as políticas da organização. Antes de reutilizar um requisito, o engenheiro de requisitos deve aferir se este ainda se mantém válido desde a última vez que foi identificado no sistema original.

É da competência do engenheiro de requisitos utilizar eficazmente e de forma produtiva as técnicas para identificar requisitos nas fontes que tem à sua disposição. O sucesso desta actividade depende essencialmente da sua experiência, capacidade de análise e preparação prévia que investe na contextualização do sistema a especificar.

Como qualquer outra actividade do ciclo de desenvolvimento de software, a descoberta de requisitos tem recursos limitados. Dentro das limitações orçamentais e temporais, devem ser dados os meios suficientes ao engenheiro de requisitos para que possa utilizar as técnicas de forma diversificada que lhe permitam obter a lista inicial de requisitos o mais completa possível. Na figura 10 relacionam-se as fontes

[2] A *Unified Modeling Language* (UML) é uma linguagem de modelação de sistemas que inclui notação gráfica.

de requisitos com as técnicas de análise em que é mais provável que sejam utilizadas. A sugestão da figura não indica, porém, que o engenheiro de requisitos não possa descobrir requisitos de outras combinações de fontes e técnicas.

Figura 10 – Aplicação de técnicas de descoberta de requisitos

2.4.2 Análise e negociação de requisitos

Análise e negociação de requisitos constituem o conjunto de actividades que se seguem à sua descoberta e identificação. Destinam-se, sobretudo, a encontrar problemas com os requisitos e chegar a acordos para os resolver. Resultam do reconhecimento de que a lista de requisitos descobertos não corresponde ainda a um conjunto completo e possui imperfeições inevitáveis da falta de uma análise integrada.

Nas actividades da análise e negociação de requisitos, o engenheiro de requisitos desempenha um papel simultaneamente de analista de sistemas e mediador dos interessados do mesmo. Nesse contexto procura resolver conflitos existentes de forma a não comprometer a qualidade do sistema.

O modelo da figura 12 representa as actividades do processo de análise e negociação.

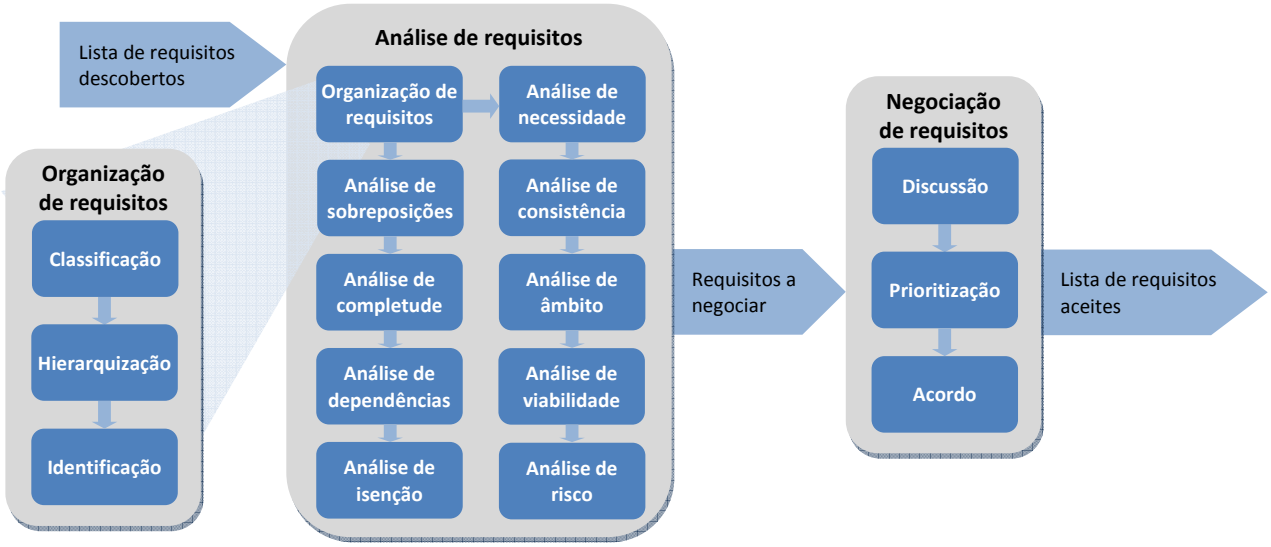

Figura 11 – Modelo das actividades de análise e negociação de requisitos

Uma preocupação do engenheiro de requisitos deve ser a de não negligenciar os requisitos não-funcionais. Estes não dizem respeito especificamente às funcionalidades de um sistema contudo podem interferir com os requisitos funcionais. Essencialmente condicionam o processo de desenvolvimento e especificam restrições externas às quais este tem de obedecer. Algumas restrições podem mesmo implicar que alguns requisitos funcionais tenham que ser abandonados.

Os requisitos do processo correspondem às restrições impostas, algumas vezes pelos próprios interessados, à forma como o processo de desenvolvimento de software deve ser conduzido. Podem incluir indicações sobre os padrões e métodos que devem ser adoptados ou impor a utilização de ferramentas CASE[3] específicas. Frequentemente são impostos por organizações de grande dimensão e que possuem normas bem definidas, com práticas de qualidade nos seus processos. As restrições podem variar no nível de detalhe desde instruções muito específicas a apenas recomendação das normas que o processo tem de adoptar, como, por exemplo, a ISO 9000[4]. Note-se que os requisitos do processo podem colocar restrições muito fortes no desenho do sistema. Por exemplo, a imposição do uso de uma ferramenta CASE por cultura organizacional pode vedar as opções de arquitectura do sistema por não suportar um paradigma de programação específico.

Tendo por base a classificação de Kotonya e Sommerville [2], os requisitos não-funcionais podem ser agrupados em três categorias de acordo com o tipo de restrição que impõem: requisitos do processo, requisitos do produto e requisitos externos, como representado na figura 12.

[3] *Computer-Aided Software Engineering* (CASE) são aplicações de apoio a actividades de Engenharia de Software.
[4] ISO 9000 é um conjunto de normas que podem ser utilizadas para certificação de qualidade.

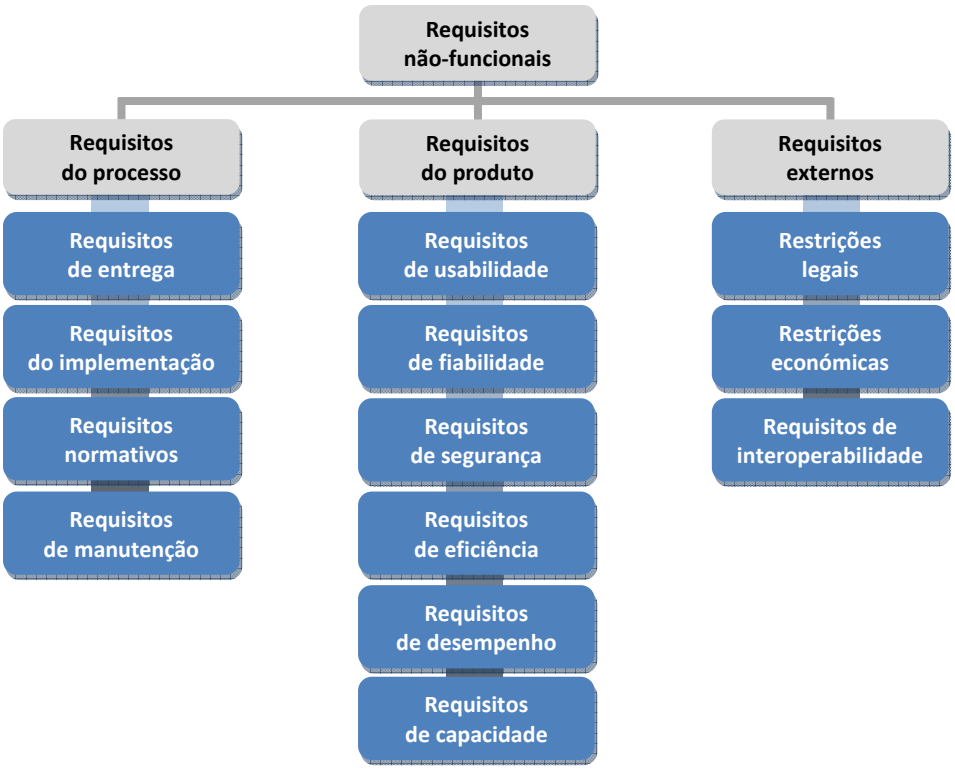

Figura 12 – Classificação dos requisitos não-funcionais

Os requisitos do produto correspondem às características que o sistema deve exibir, sendo habitualmente designados de propriedades do sistema. Algumas, como as que dizem respeito a desempenho e capacidade, podem ser formuladas de modo preciso e são facilmente quantificáveis. Outras, como a usabilidade, são mais difíceis de quantificar e, por isso, são expressas de forma mais informal. A maioria dos requisitos de produtos concentra-se em especificar restrições ao comportamento do sistema no contexto da sua colocação em produção. Tal como os requisitos do processo, limitam a liberdade da equipa de desenho do sistema. Os requisitos não-funcionais deste tipo podem facilmente entrar em conflito entre si. Por exemplo, um requisito que imponha um certo nível de desempenho pode ser contrariado por outros que imponham elevados níveis de fiabilidade e segurança. O nível de importância de cada requisito será determinante para se atingir um compromisso que permita ultrapassar o conflito, frequentemente com o desfecho de sacrificar os requisitos considerados menos importantes em detrimento dos que melhor se alinham com os objectivos gerais do sistema.

Os requisitos externos são na realidade requisitos que podem ser colocados tanto no produto como no processo mas que provêm sempre do ambiente externo à organização. Podem ser baseados em informação do domínio de aplicação ou considerações organizacionais ou em necessidades de comunicação com outros sistemas ou regulamentos de protecção de dados.

A norma ISO 9126 define as características recomendadas para avaliar a qualidade de um produto de software. As características são apresentadas como um conjunto de atributos, designadamente: funcionalidade, fiabilidade, usabilidade, eficiência, facilidade de manutenção e portabilidade [9]. Verifica-se que a classificação de requisitos da figura 12 cobre os atributos de qualidade enumerados pela norma.

Sem perder de vista os requisitos não-funcionais, a análise de requisitos engloba essencialmente dois conjuntos de actividades: organização dos requisitos descobertos e análise crítica sobre os mesmos.

A organização de requisitos é uma actividade previsível que permite torná-los geríveis e é fundamental para a sua análise. Engloba actividades de:

- Classificação – Implica classificar os requisitos em categorias de acordo com o seu tipo: funcionais e não-funcionais, criar sub-categorias dos requisitos não-funcionais de acordo com os critérios apresentados na figura 12 e criar sub-categorias dos requisitos funcionais por grupos de funcionalidades.
- Hierarquização – Implica criar relações de composição entre os requisitos de acordo com o seu nível de abstracção, colocando os pouco detalhados nos níveis superiores da hierarquia e os muito detalhados nos níveis inferiores.
- Identificação – Implica atribuir um identificador único a cada requisito de forma a poder referenciá-lo. O identificador pode ser um número sequencial ou seguir uma lógica que reflicta a organização em categorias e/ou a hierarquia.

A análise crítica é uma actividade que depende da experiência do engenheiro de requisitos e da sua capacidade para detectar problemas. Nos tipos de análise seguintes, o engenheiro de requisitos tem autonomia suficiente para realizar acções correctivas:

- Análise de completude – Implica analisar se as descrições dos requisitos são completas, ou seja, se não omitem informação importante para que possam ser compreendidos pelas equipas que vão realizar as etapas seguintes do ciclo de desenvolvimento de software. Se não forem considerados completos, a sua descrição terá que ser desenvolvida de forma a incluir a informação em falta.
- Análise de dependências – Implica identificar e registar as dependências que existem entre os requisitos. Na descrição de cada requisito, devem ser mantidas referências a outros requisitos dos quais esse dependa. Podem ser utilizadas técnicas para sistematizar esta análise tais como matrizes de dependências implementadas em folhas de cálculo.
- Análise de sobreposições – Implica analisar se existem requisitos cujas descrições se repetem ou que cobrem a mesma informação. As partes não-

comuns devem permanecer em cada requisito. A informação comum deve ser isolada e abstraída para um requisito próprio. Se houver necessidade, o requisito abstraído pode ser referenciado por outros.

- Análise de isenção – Implica avaliar se os requisitos funcionais são isentos de detalhes de desenho (lógico ou físico) e implementação. Um requisito funcional deve descrever o que o sistema deve fazer e, sempre que possível, estar desprovido de detalhes que possam influenciar a forma de o implementar. Se esses detalhes forem encontrados, deve ser discutida a necessidade da sua inclusão.

Nos tipos de análise seguintes, os problemas identificados terão que ser remetidos para a negociação onde serão discutidos, prioritizados e acordados pelos interessados do sistema com poder de decisão para tal:

- Análise de necessidade – Implica identificar quais são os requisitos propostos que não contribuem para os objectivos de negócio da organização. Na negociação poderão ser reformulados para melhor ser percepcionada a sua relevância ou rejeitados.
- Análise de consistência – Implica identificar contradições entre os requisitos que poderão levar a conflitos na sua implementação. Na negociação terão que ser resolvidos por intermédio de acordos.
- Análise de âmbito – Implica delimitar a fronteira de actuação do sistema a desenvolver e seleccionar se cada requisito está dentro ou fora do sistema. Na negociação terá de ser decidido o que fazer aos requisitos que estão fora do sistema: se devem ser rejeitados ou incluídos, implicando nesse caso uma redefinição da fronteira.
- Análise de viabilidade – Implica avaliar se os requisitos podem ser implementados dentro das restrições orçamentais e respeitando os prazos para a execução do projecto. Identifica dessa forma os requisitos considerados inviáveis à luz dessas restrições. Na negociação terá de ser decidido o que fazer a esses requisitos: aumentar os recursos disponíveis ou rejeitá-los. Poderá também ser decidido implementá-los em versões posteriores do sistema.
- Análise de risco – Implica identificar as dificuldades em analisar alguns requisitos e antecipar os obstáculos que possam surgir no desenvolvimento. A análise pode incidir em tipos de risco específicos: técnico, desempenho, segurança, integridade, etc. Na negociação estes requisitos terão que ser discutidos e medido o impacto que pode ter a sua implementação.

As actividades que constituem a negociação de requisitos são:

- Discussão – Os interessados discutem os requisitos que foram destacados como problemáticos. Cada interessado apresenta a sua perspectiva sobre cada requisito [10]. É discutida a necessidade de cada requisito e são debatidos os requisitos contraditórios e os conflitos que introduzem. Decide-se que requisitos fazem parte do sistema, e em algumas situações, reconsidera-se a fronteira do sistema para que inclua requisitos que anteriormente estariam fora. Equaciona-se igualmente a viabilidade e o risco de alguns requisitos. Desta discussão, alguns requisitos serão imediatamente rejeitados enquanto outros poderão passar à actividade seguinte.
- Prioritização – São atribuídas prioridades aos requisitos de acordo com escalas quantitativas ou qualitativas previamente definidas. Os requisitos com mais elevada prioridade serão implementados na primeira versão do sistema enquanto os outros serão remetidos para versões futuras.
- Acordo – Identificam-se as opções para solucionar os problemas com os requisitos que estiveram em discussão. Algumas podem ser consensuais enquanto outras resultam de compromissos entre os vários interessados. Os compromissos implicam cedências e contrapartidas entre os interessados, sendo essenciais para servirem de base a acordos. Alguns requisitos podem ter que ser alterados para acomodarem o compromisso.

Em alguns casos, o processo de análise levanta questões que não podem ser respondidas e para as quais não há acordo possível. De uma forma geral, tal denuncia que não existe informação suficiente disponível para a negociação. No modelo em espiral da figura 8, essa situação corresponde a uma nova iteração para recolher mais informação nas áreas críticas. Caso se chegue a acordo sobre os requisitos com máxima prioridade é produzida uma lista dos requisitos estáveis e aceites por todos os interessados. Esta lista contém os requisitos que podem ser incluídos no documento de especificação de requisitos.

2.4.3 Especificação e documentação de requisitos

A actividade de especificação e documentação de requisitos destina-se a produzir um esboço do documento de especificação. Os requisitos na lista de requisitos aceites pelas actividades de análise e negociação serão alvo de uma descrição rigorosa e oficial a incluir nesse documento.

O objectivo primordial do documento de especificação é comunicar o que se pretende que um sistema faça. Destina-se a todos os interessados no sistema, sejam clientes, utilizadores finais ou equipas de desenvolvimento. Tanto pode assumir a forma de um documento sucinto e genérico, nas primeiras iterações do ciclo de desenvolvimento de software, como pode ser um documento extenso e muito

detalhado quando for para ser utilizado pelas equipas de desenvolvimento. Como se verifica na figura 8, o documento vai aumentando em cada iteração à medida que aumentam também as saídas das actividades anteriores do processo de desenvolvimento.

A escrita do documento de especificação cabe directamente ao engenheiro de requisitos. Este deve possuir competências em comunicação [10] e ser sensível a aspectos de usabilidade já que são também essenciais para realizar as actividades de descoberta e análise de requisitos. Deve também garantir que, sempre que possível, não sejam incluídos aspectos que interfiram com a liberdade de actuação das equipas de desenho e implementação.

Um documento de especificação de requisitos é escrito em linguagem natural, podendo ser complementado com diagramas, tabelas, fotografias e imagens sempre que tal seja útil para clarificar a informação que se pretende transmitir. Existem recomendações que sugerem qual a informação que deve estar presente em documentos de especificação de requisitos. De uma forma geral, o documento deve incluir [2]:

- A visão geral da organização, do processo de negócio suportado pelo sistema e dos benefícios decorrentes do seu desenvolvimento;
- Definições de todos os serviços que estejam associados aos requisitos funcionais do sistema;
- Definições de todas as propriedades que estejam associadas aos requisitos não-funcionais do sistema;
- Restrições à operação do sistema e ao seu processo de desenvolvimento;
- Descrição do ambiente em que o sistema será colocado em produção e as alterações que o sistema poderá impor no mesmo;
- Especificações detalhadas de todos os requisitos, apoiadas por linguagem formal, modelos ou outras ferramentas de análise;
- Um glossário que explique termos específicos como apêndice do documento.

Esta informação pode ser organizada num documento de vários modos uma vez que existem várias formas de estruturar um documento de especificação de requisitos. Um exemplo é a estrutura proposta pela IEEE 830-1998[5] [11], apresentada na figura 13. Além da estrutura, inclui também um conjunto de recomendações de como devem ser escritos os requisitos de forma a evitar alguns erros comuns. É, por isso, considerada um bom ponto de partida para definir a sua estrutura. Os primeiros dois capítulos contextualizam o documento e o sistema de forma muito genérica. O terceiro capítulo descreve os requisitos de forma detalhada.

[5] IEEE 830-1998 é uma norma de recomendações práticas para especificações de requisitos de software.

As suas secções reflectem a classificação efectuada na sua organização. Será aquele que sofrerá sucessivas revisões ao longo do processo de engenharia de requisitos.

1. Introdução
   - 1.1 Propósito do documento
   - 1.2 Âmbito do sistema
   - 1.3 Definições, acrónimos e abreviaturas
   - 1.4 Referências
   - 1.5 Resenha do resto do documento
2. Descrição geral
   - 2.1 Perspectiva do produto
   - 2.2 Funções do produto
   - 2.3 Características dos utilizadores
   - 2.4 Restrições gerais
   - 2.5 Pré-condições e dependências
3. Requisitos específicos
   - Enumeração e descrição detalhada dos requisitos funcionais e requisitos não-funcionais

Figura 13 – Modelo do documento de requisitos da norma IEEE 830-1998

Outras visões do documento, como a de McConnell [5], sugerem que devem existir dois documentos para descrever os requisitos em vez de um único. Deve haver um manual de utilizador preliminar e um protótipo da interface com o utilizador que tem como público-alvo todos os interessados e potenciais utilizadores do sistema. O outro é um documento formal de requisitos que é considerado um complemento. Este é especialmente dirigido às equipas de desenvolvimento, implementação e qualidade.

Independentemente do formato escolhido para o documento de especificação de requisitos, deve ser garantida a sua legibilidade pelos interessados. Nesse sentido, pode considerar-se que existem duas formas de escrever requisitos, utilizado:

- Linguagem natural – Textos escritos numa língua comum a todos os interessados com descrições o mais completas possível. As descrições podem ser apoiadas por diagramas UML contudo estes nunca as podem substituir. Alguns exemplos de frases encontradas em requisitos escritos em linguagem natural são "o sistema deve ter uma disponibilidade superior a 90%", "o sistema deve respeitar as normas de acessibilidade do W3C" ou "o sistema deve ter um controlo de acessos com autenticação".
- Linguagem formal – Expressões com uma sintaxe fortemente estruturada que seguem uma especificação formal. Alguns exemplos de sintaxes com especificações formais são: fórmulas matemáticas para especificar cálculos, expressões EBNF[6] para especificar linguagens de programação, expressões

[6] *Extended Backus–Naur Form* (EBNF) é uma notação para especificar gramáticas livres de contexto.

regulares para especificar restrições a gamas de valores, XML Schema[7] para especificar a estrutura, conteúdo e semântica de documentos ou expressões OCL[8] para especificar restrições em modelos UML.

A utilização de qualquer das linguagens de descrição tem vantagens e inconvenientes. A linguagem natural tem a vantagem de ser de fácil compreensão por todo o público-alvo do documento. Seria impossível tentar descrever um sistema somente com recurso a linguagem formal. Ainda que a natureza do sistema o permitisse, o público-alvo seria demasiado restrito para a tornar útil. Só os profissionais da área de negócio do sistema estariam em condições de compreender o formalismo utilizado tornando-os incompreensível para o restante público-alvo.

A tabela 1 compara as características mais importantes da utilização da linguagem natural e da linguagem formal.

Tabela 1 – Atributos de linguagens de escritas de requisitos

Atributo	Descrição utilizando linguagem natural	Descrição utilizando linguagem formal
Barreira lexical	Exige domínio da língua utilizada	Exige conhecimento da sintaxe da linguagem formal
Compreensão	Fácil por todo o público-alvo	Restrita aos profissionais da área de negócio e da equipa de desenvolvimento
Precisão	Baixa	Elevada
Ambiguidade	Provável	Improvável
Testabilidade	Baixa	Elevada

É consensual que os requisitos devem ser sempre escritos em linguagem natural para promover a sua compreensão por todo o público-alvo. Contudo, verifica-se que a simples utilização de linguagem natural não é suficiente. É necessário que seja clara e o menos sujeita a ambiguidades possível. A utilização exclusiva de linguagem natural para descrever requisitos, mesmo com o cuidado de não introduzir ambiguidade, contínua a ser passível de causar incerteza nos leitores através de diferentes interpretações da mesma frase. A estruturação de frases, a escolha de termos ou a apresentação de raciocínios podem influenciar a precisão de uma descrição. Para resolver de forma definitiva essa incerteza, a descrição deve ser complementada com expressões em linguagem formal. A grande vantagem da utilização de linguagem formal reside na precisão com que é especificado o requisito. Ao utilizar sintaxes fortemente estruturadas não restam possibilidades para a existência de ambiguidade porque só é possível uma e uma só interpretação da expressão. Frequentemente, a linguagem formal é utilizada tendo em mente a equipa

[7] XML Schema é uma linguagem de especificação de esquemas da *Extensible Markup Language* (XML) que se tornou recomendação do *World Wide Web Consortium* (W3C).

[8] *Object Constraint Language* (OCL) é uma linguagem declarativa para especificar restrições a modelos UML.

de desenvolvimento e permite aliviar as descrições textuais que seriam excessivamente densas para obter o mesmo nível de precisão.

Os diagramas em UML são uma forma muito clara e flexível de estabelecer os elos de ligação entre as descrições em linguagem natural e linguagem formal. Um diagrama nunca dispensa uma descrição textual mas é possível incluir expressões formais junto aos elementos a que dizem respeito.

2.4.4 Validação de requisitos

As actividades de validação de requisitos destinam-se a avaliar o documento de especificação de requisitos e aprová-lo de forma a poder servir os seus propósitos: permitir estabelecer uma base de acordo entre os interessados, fornecer documentação às restantes etapas do ciclo de desenvolvimento de software, promover a compreensão do negócio na organização e apoiar a gestão de projectos.

Pretende-se obter uma versão estável do documento de especificação de requisitos a partir do esboço produzido na fase anterior. Essa versão será a utilizada nas etapas de desenho lógico e físico, implementação e codificação e verificação e validação do ciclo de desenvolvimento de software.

A validação de requisitos está relacionada com a verificação de atributos dos mesmos já na sua forma oficial, ou seja, como descrições no documento de requisitos. Numa primeira observação, poderá parecer uma tarefa redundante, uma vez que nas actividades de análise já foram efectuadas verificações. A diferença situa-se nos objectivos das duas verificações. Enquanto as verificações da análise identificam um conjunto de problemas nos requisitos para garantir que o conjunto dos aceites são os certos para o sistema, as verificações da validação de requisitos procuram garantir que, na sua descrição oficial, estes estão bem documentados.

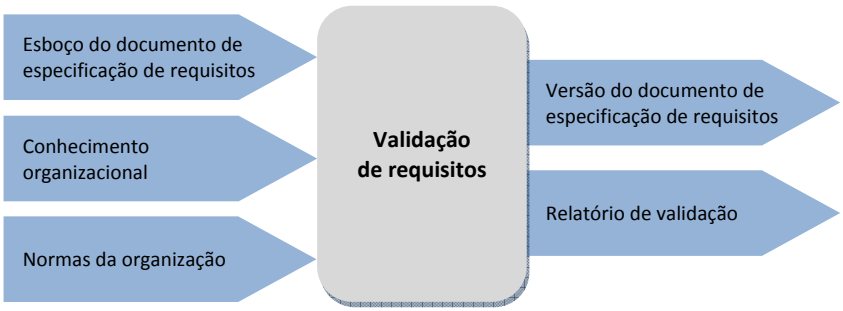

Figura 14 – Modelo da validação de requisitos

Ao contrário do processo de análise de requisitos, a validação do documento de especicação é efectuada por um conjunto dos interessados que tenham bons conhecimentos das normas da organização e que conheçam bem a própria organização. Estas normas e o esboço do documento de requisitos constituem as entradas do processo. As saídas do processo são uma versão final do documento de

especificação de requisitos e um relatório de validação. O modelo do processo encontra-se representado na figura 14.

No relatório de validação são indicadas duas listas [2]:

- Lista de problemas – Lista com os problemas encontrados no próprio documento de especificação de requisitos. Deve estar organizada por tipo de problema (ex: problemas de ambiguidade, problemas de falta de consistência no documento, etc.) porém verifica-se que, na prática, não é fácil atribuir cada problema a um só tipo.
- Lista de acções acordadas – Lista das acções a realizar para dar resposta aos problemas da lista anterior. Tal como na negociação de requisitos, as acções a efectuar resultam de acordo entre os interessados responsáveis pela validação. Pode não existir uma correspondência unívoca entre os problemas e as acções correctivas. Alguns problemas podem necessitar de várias acções correctivas para serem ultrapassadas. Da mesma forma, a mesma correcção pode resolver mais do que um problema.

A verificação de requisitos é um processo demorado. Implica que um conjunto de pessoas reflicta sobre um documento que pode ser bastante longo. Se o número de pessoas for muito elevado e o sistema a desenvolver for complexo podem decorrer alguns meses até que a validação seja concluída. À medida que as pressões dos prazos se intensificam, há uma tendência natural para tentar apressar o processo para que seja possível o início das etapas de desenho e implementação. Segundo Kotonya e Sommerville, um correcto planeamento inicial da validação do documento de requisitos permite que essa situação não ocorra. Se o processo for apressado, incorre-se nos riscos já identificados e associados a uma especificação de requisitos deficiente com problemas cujos custos de correcção serão mais elevados, como representado na figura 4.

As principais técnicas de validação de requisitos são a revisão, a prototipagem e a concepção de testes. A revisão é a mais utilizada porque pode ser realizada por qualquer elemento da equipa de validação. A prototipagem envolve aproveitar o protótipo utilizado na descoberta e identificação de requisitos, caso este tenha sido desenvolvido. A concepção de testes poderá antecipar a existência de problemas. Note-se que os testes só são efectivamente realizados na etapa do ciclo de desenvolvimento de software.

A equipa de validação baseia o seu trabalho na avaliação dos atributos de qualidade dos requisitos indicados na tabela 2 [5], mediante a realização de reuniões de revisão.

Tabela 2 – Atributos dos requisitos

Atributo	Descrição
Compreensão	Indica a facilidade de percepção dos requisitos na sua descrição.
Ambiguidade	Indica se os requisitos se prestam a interpretações diferentes por cada interessado.
Correcção	Indica se os requisitos são realistas, verdadeiros e exequíveis.
Rastreabilidade	Indica se os requisitos referem as razões que levaram à sua inclusão.
Consistência	Indica se existem contradições na informação contida nos requisitos entre si.
Conformidade	Indica se os requisitos aderem às normas definidas para a sua escrita.
Relevância	Indica a necessidade de referir o requisito no contexto geral do documento de especificação.
Organização	Indica a forma como os requisitos estão organizados.
Redundância	Indica se a mesma informação se repete em mais do que um requisito.
Manutenção	Indica se a estrutura lógica do documento facilita inevitáveis alterações que ocorrem.
Completude	Indica se o requisito contém toda a informação necessária à sua implementação.
Testabilidade	Indica se é o requisito inclui informação que permita validar o seu cumprimento.

Todos os atributos, excepto os de ambiguidade e redundância, contribuem para a boa qualidade do documento de requisitos.

As acções de correcção mais comuns a efectuar, decorrentes, das reuniões são:

- Reformulação – Os requisitos que não estão escritos na forma mais correcta, de interpretação difícil ou pouco clara têm que ser reformulados. É necessário reescreve-los de forma a melhorar a sua compreensão. As recomendações são:
  - Utilizar linguagem clara, precisa e directa;
  - Evitar frases longas;
  - Utilizar termos especializados mas remeter a sua definição para o glossário;
  - Enriquecer a descrição com diagramas UML;
  - Organizar informação relacionada em tabelas;
  - Complementar a descrição utilizando linguagem formal.

  Estas recomendações permitem não só aumentar a compreensão dos requisitos como eliminar eventuais ambiguidades que ainda permaneçam.
- Aperfeiçoamento – Caso se verifique falta de informação no documento de especificação de requisitos, é necessário encontrar mais informação nas fontes de requisitos ou incluir informação já encontrada mas omissa no documento. As recomendações são:
  - Cobrir mais cenários nos requisitos incompletos;
  - Incluir casos de teste;
  - Incluir medidas quantitativas na descrição do requisito;
  - Ligar cada requisito a outros de que dependa.

Estas recomendações permitem obter mais informação para cada requisito, reduzir a redundância de informação e aumentar o seu nível de testabilidade.

- Avaliação – Se subsistirem dúvidas quanto à correcção da informação de alguns requisitos que não permitem aferir a sua viabilidade. As recomendações são:
  - Garantir a autenticidade do que é descrito no requisito;
  - Confirmar a exequibilidade do requisito;
  - Ligar cada requisito à sua fonte.

  Estas recomendações permitem aumentar a correcção e simultaneamente aumentar a rastreabilidade e reforçar a relevância dos requisitos.
- Uniformização – Quando existirem inconsistências na forma como os requisitos são documentados. As recomendações são:
  - Eliminar inconsistências entre os requisitos que tenham permanecido da negociação;
  - Uniformizar o estilo de escrita de forma a seguir sempre as normas;
  - Uniformizar terminologia evitando sinónimos.

  Estas recomendações permitem aumentar o nível de consistência de todo o documento e aumentar a conformidade com as normas da organização.
- Reorganização – Se forem detectados problemas na forma como o documento é organizado então a organização por níveis e categorias, efectuada na análise de requisitos, tem que ser revista. As recomendações são:
  - Mudar requisitos de nível;
  - Mudar requisitos de categorias.

  Estas recomendações permitem melhorar a organização, facilitando a manutenção do documento.

O relatório de verificação deve ser utilizado como referência para a próxima iteração do processo em cada etapa subsequente.

Para além da revisão de requisitos, a prototipagem também pode ser uma importante técnica de validação dos mesmos [2]. Os protótipos usados num projecto para identificação de requisitos podem ser reutilizados para a sua validação. Porém, existem diferenças na sua utilização nas duas fases. Um protótipo concebido para a descoberta de requisitos deve incluir somente os requisitos que são de compreensão particularmente difícil de descrever ou compreender, podendo omitir os que são facilmente compreensíveis. Um protótipo utilizado para validação terá que ser um sistema utilizável de uma forma natural, não podendo omitir as funcionalidades mais básicas. Tem que implementar um subconjunto dos requisitos

aceites enquanto que o protótipo inicial pode implementar requisitos não aprovados.

A construção de um protótipo inicia-se pela selecção das pessoas que o vão utilizar. A escolha deve recair nos utilizadores com maior abertura na utilização de um novo sistema e, se possível, das várias áreas de negócio incluídas no sistema. De seguida, são desenvolvidos cenários de teste que cubram o melhor possível os requisitos, atendendo ao tempo que os utilizadores dispõem para realizar estes testes. Os utilizadores são deixados com o protótipo para executar cenários de uma utilização realista do sistema. O engenheiro de requisitos não pode interferir com esse processo. Deve observar a utilização sem dar recomendações. Desta observação podem ser revelados detalhes importantes sobre o modo como os utilizadores encaram as falhas no sistema ou problemas com a sua utilização. Estes detalhes devem ser documentados no relatório de validação.

Se os utilizadores forem informados do interesse da sua utilização, a prototipagem pode ser uma técnica importante de validação. É importante realçar aos utilizadores que não se trata do sistema final e identificar quais são as funcionalidades ainda não implementadas, para garantir que os objectivos da sua utilização são atingidos.

Outra técnica importante de validação, e que pode ser utilizada como complemento das revisões, é a concepção de testes para aferir a sua correcta implementação [2]. Um requisito com boa testabilidade permite definir um ou mais testes que verifiquem o seu cumprimento no sistema. Embora os testes sejam concebidos na validação de requisitos, só serão executados na etapa de verificação e validação do produto do ciclo de desenvolvimento de software. A abordagem a adoptar é a de *test-driven specification* em que os casos de teste são concebidos durante a especificação de requisitos, tratando-se de uma variante à abordagem de *test-driven development* , em que os casos de teste são criados imediatamente antes da codificação de cada requisito durante a implementação e codificação [12].

A técnica consiste em analisar o documento de especificação e definir para cada requisito um ou mais casos de teste. A sua proliferação, contudo, deve ser evitada porque dificulta a sua manutenção quando forem codificados. Para manter um número mínimo de casos de teste deve tentar-se que cada um seja concebido para abranger vários requisitos, criando classes de equivalência. Se for necessário utilizar muitos casos de teste para cobrir um mesmo requisito, tal implica que a organização dos requisitos deve ser revista de forma a decompô-lo em sub-requisitos.

Para cada requisito testado deve ser preenchido um registo de teste que será utilizado posteriormente na etapa de verificação e validação. O registo de teste deve incluir a seguinte informação:

- Identificador do requisito – Referente atribuído na fase de análise;

- Requisitos relacionados – Identificação de outros requisitos que o teste também cubra;
- Descrição do teste – Indicação de como é realizado o teste, quais os seus objectivos, entradas e os resultados esperados.
- Problemas na realização do teste – Descrição do problema no requisito que torna a concepção de um teste difícil ou mesmo impossível.
- Comentários e recomendações – Conselhos de como poderão ser resolvidos os problemas nos requisitos de forma a tornar possível a sua testabilidade.

Na etapa de verificação e validação do ciclo de desenvolvimento de software, os resultados esperados serão comparados com os resultados obtidos.

A concepção de testes no contexto da validação de requisitos é uma abordagem que permite antecipar muitos problemas que não são facilmente perceptíveis por simples inspecção do documento de especificação. Contribui também para a redução dos custos de manutenção porque transfere a resolução desses problemas para a etapa de especificação onde esse custo ainda é bastante baixo.

2.5 O processo de gestão de requisitos

As alterações aos requisitos são uma realidade inevitável em qualquer projecto de software e podem surgir tanto durante o seu desenvolvimento como após este ser colocado em produção. Contemplar alterações aos requisitos constitui sempre um problema para as equipas de desenvolvimento. A gestão de requisitos surge para minimizar esse transtorno com actividades de controlo e documentação dessas alterações.

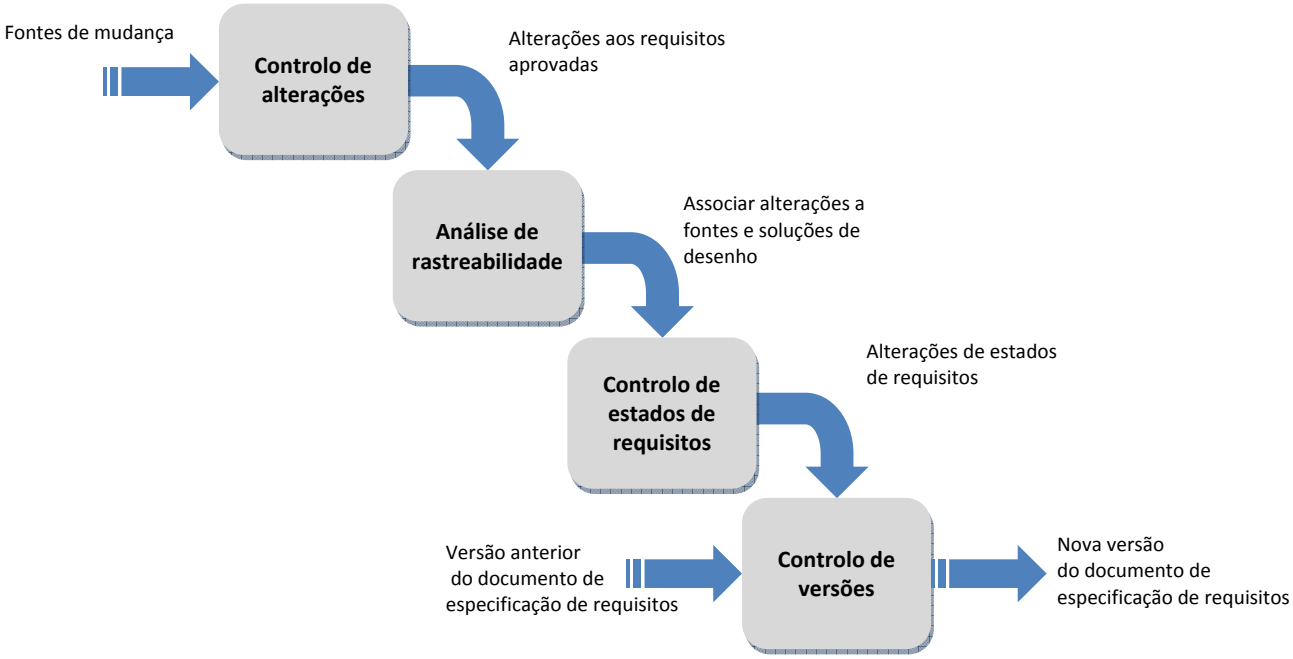

Figura 15 – Modelo do processo de gestão de requisitos

O modelo representado na figura 15 procura identificar e sequenciar as quatro actividades do processo de gestão de requisitos. Na prática, verifica-se que este processo não é tão passível de sequenciação como o de desenvolvimento de requisitos.

É possível que as actividades sejam efectuadas em simultâneo, Contudo, numa óptica de modelo em cascata em que as saídas de uma actividade são as entradas da seguinte, o modelo pode ser encarado como ponto de partida, de forma a servir de base a uma aplicação apropriada para cada organização.

O processo de gestão de requisitos inclui várias actividades:

- Controlo de alterações – Análise das propostas de alterações de requisitos e avaliação do impacto que cada alteração terá no desenvolvimento antes de a aprovar. Os novos compromissos são negociados a partir da estimativa de impacto das alterações. As alterações aprovadas são incorporadas nos requisitos de uma forma controlada.
- Análise de rastreabilidade – A rastreabilidade é um atributo essencial nos requisitos para permitir a sua gestão eficaz. Permite aferir o motivo que justifica a existência de um requisito, ligando-o à fonte que lhe deu origem, e quais os que lhe estão relacionados. Tal é fundamental para se compreender o impacto que uma alteração num requisito pode ter nos restantes.
- Controlo de requisitos – No seu ciclo de vida, um requisito pode transitar por vários estados. O estado de cada requisito é controlado e é mantido um histórico dos vários estados que já teve.
- Controlo de versões – O documento de requisitos é mantido actualizado reflectindo as alterações aprovadas. O desenho e implementação são planeados de forma a efectuar várias alterações numa única versão.

Existem ferramentas CASE para apoiar o processo de gestão de requisitos. Algumas organizações produzem as suas próprias ferramentas para suportarem o seu processo. As características mais relevantes destas ferramentas são:

- Armazenamento de requisitos num SGBD[9], com suporte da classificação dos requisitos funcionais e não-funcionais, mapeamento em módulos do sistema e organização hierárquica dos requisitos.
- Apoio à produção de uma versão do documento de especificação com geração automática de uma versão do documento a partir dos requisitos aprovados e de outras informações extraídas de uma base de dados.

[9] Sistemas de Gestão de Bases de Dados (SGBD) são sistemas que oferecem funcionalidades de acesso, manutenção e gestão de bases de dados.

- Apoio ao controlo de requisitos com registo dos estados de cada um, alteração controlada do seu estado e consulta de histórico com associação a versões do documento de especificação.
- Apoio à análise de rastreabilidade, com registo das relações dos requisitos entre si entre a sua informação de rastreabilidade, e inclusão automática no documento de especificação.

2.5.1 Controlo de alterações

O impacto de qualquer alteração deve ser cuidadosamente avaliado antes de ter repercussões nos requisitos já existentes e na implementação do produto. Caso seja aprovada, deve ser planeada a sua implementação e, se possível, integrada com outras alterações para minimizar o número de versões do produto. Se não houver este tipo de controlo pode correr-se o risco de serem aprovadas alterações de baixa prioridade antes de outras de maior prioridade. Outra consequência surge se forem implementadas alterações, o que pode vir a revelar-se dispendioso, sem antes haver uma reflexão sobre a sua relevância.

A actividade de controlo de alterações tem como objectivos:

- Recolher alterações das fontes de mudança e daí derivar alterações aos requisitos;
- Analisar o impacto dessas alterações;
- Propor alterações aos requisitos junto dos interessados do sistema;
- Obter decisões dos interessados sobre essas alterações;
- Alterar a descrição dos requisitos;
- Medir a volatilidade de requisitos.

As fontes de mudança são todos os eventos que fazem pressão para que ocorram alterações aos requisitos do sistema [2]. A tabela 3 identifica essas fontes com uma breve descrição de cada uma.

Embora as alterações sejam inevitáveis, é possível identificar alguns requisitos mais estáveis que outros. Os mais estáveis são os que dizem respeito ao domínio de aplicação do sistema, estando mais protegidos de alterações e designando-se de requisitos não-voláteis. Embora possam, ocasionalmente, necessitar de ser alterados, essas ocorrências são raras durante o ciclo de vida do sistema. Os requisitos menos estáveis, mais passíveis de alterações, designam-se de requisitos voláteis. Estes requisitos são normalmente específicos das necessidades de um sistema para uma organização em particular e, frequentemente, para um ambiente em particular.

Tabela 3 – Fontes de mudança dos requisitos

Fontes de mudança	Descrição
Erros, conflitos e inconsistências nos requisitos	À medida que os requisitos são analisados e implementados, vão surgindo erros e inconsistências que têm que ser corrigidas. Alguns podem ser descobertos durante a análise ou na validação de requisitos. Outros só o são posteriormente, implicando alterações para as correcções.
Aumento do conhecimento do sistema	À medida que os requisitos vão sendo descritos, os interessados compreendem melhor o que pretendem do sistema e das suas reais potencialidades, o que leva a que os requisitos sejam reformulados.
Problemas inesperados	Podem ser encontrados problemas técnicos, de prazos ou orçamentais na implementação de alguns requisitos. Pode ser mais dispendioso ou levar mais tempo do que inicialmente previsto. Os requisitos têm que sofrer alterações para resolver estes problemas.
Redefinição das prioridades do cliente	As prioridades do cliente alteram-se durante o desenvolvimento do sistema em resposta a alterações no ambiente do negócio como o aumento da concorrência ou diminuição de recursos humanos. Essas alterações interferem com a prioridade de implementação dos requisitos atribuída na sua negociação.
Alterações no ambiente operacional	Os pressupostos sobre o ambiente em que o sistema será colocado em produção podem ser alterados de tal forma que implique alterações aos requisitos para assegurar a compatibilidade.
Alterações organizacionais	A organização que irá utilizar o sistema pode ter alterado a sua estrutura e/ou processos, resultando em novos requisitos de sistema.
Alterações no ambiente externo	As normas, regulamentos e legislação do domínio de aplicação são alteradas por agentes externos à organização. Os requisitos têm de ser alterados para as acomodarem.

Os requisitos voláteis, de acordo com a associação às fontes de mudança e como representado na figura 16, podem ser classificados em:

- Requisitos de consequência – Baseiam-se em pressupostos sobre a forma como o sistema será colocado em produção. Quando este é operacionalizado, pode verificar-se que alguns desses pressupostos não estão correctos ou já não permanecem válidos. A adaptação dos utilizadores leva a novas formas de utilizar as suas funcionalidades, surgindo daí pedidos de alterações. Podem estar ligados a erros e inconsistências, a alterações organizacionais ou a problemas técnicos inesperados.
- Requisitos emergentes – São consequência directa do aumento do conhecimento do cliente sobre o sistema a construir. Os clientes começam a pensar em novas formas de extrair e apresentar informação do sistema assim que começam a ver exemplos de apresentação possíveis. São alterações que podem ser antecipadas por um engenheiro de requisitos experiente.
- Requisitos mutáveis – Surgem em sequência de alterações ao ambiente no qual o sistema opera. Por exemplo, a alteração de legislação força à mudança do requisito que lhe está associado. Por outro lado, também a alteração das prioridades impostas ao cliente poderá colocar pressão em mudar os requisitos bem como alterações na própria organização. São alterações normalmente difíceis de prever.

- Requisitos de compatibilidade – Estão associados à manutenção da compatibilidade do sistema. São requisitos frequentemente sujeitos a alterações visto estarem ligados a fontes fora do controlo da organização.

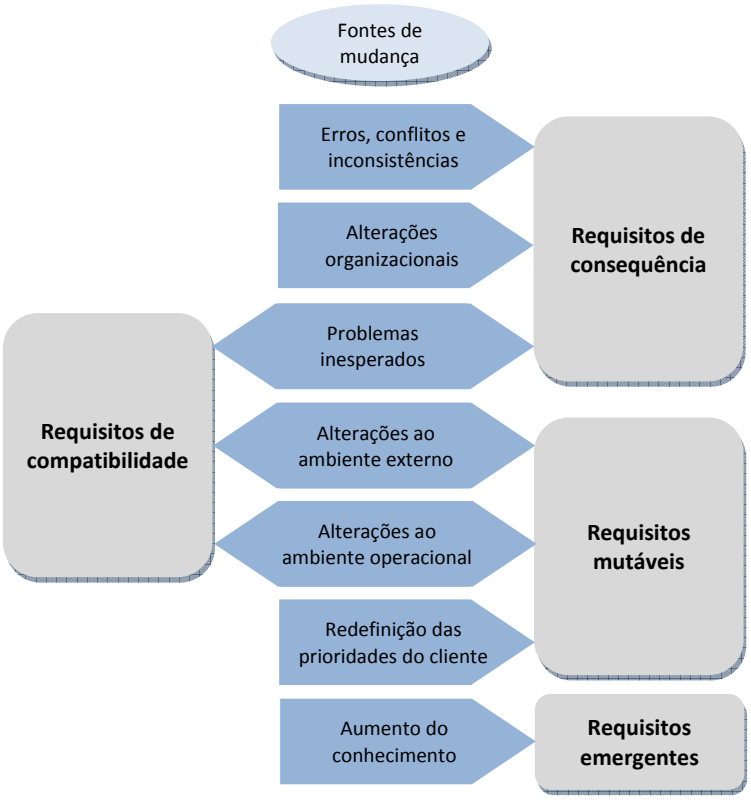

Figura 16 – Tipos de requisitos voláteis

É reconhecida como boa prática de gestão de requisitos tentar antecipar quais serão os que estarão mais sujeitos a alterações e mesmo tentar prever quais seriam essas alterações. Tal envolve classificar os requisitos, identificar os mais voláteis e prever que aspectos na sua descrição poderão mudar. A grande vantagem desse exercício é orientar o trabalho das equipas de desenvolvimento de modo a que estas possam implementar esses requisitos em componentes relativamente independentes. Por conseguinte, se essas alterações realmente se verificarem, o seu impacto no resto do sistema é diminuído.

2.5.2 Análise de rastreabilidade

Uma actividade crítica no processo de gestão de requisitos é a avaliação do impacto que uma alteração pode ter em todo o sistema. Se a alteração ocorrer enquanto os requisitos estão a ser descobertos, é necessário apenas aferir de que forma vai interferir nos demais. Se a alteração ocorrer já durante a implementação do sistema, a avaliação do seu impacto interfere não só com outros requisitos, como também com o desenho do sistema e a sua implementação. Se a alteração ocorrer

após o sistema ser colocado em produção, é necessário compreender também de que forma interfere com as actividades normais dos utilizadores.

Para que esta análise possa ser possível, é necessário dispor de informação sobre as dependências e ligações entre a definição de um requisito e a sua implementação. Este tipo de informação designa-se de informação de rastreabilidade. A alteração proposta deve ser analisada no contexto da informação extraída das fontes durante a sua descoberta. Só é possível avaliar o impacto de uma alteração se houver relação directa entre um requisito e a forma como é desenhado e implementado. Também as relações de dependência entre os requisitos são importantes para medir as interferências entre si.

Tratando-se de diferentes formas de rastreabilidade, é possível classificá-las em cinco tipos que evidenciam o sentido de navegação da informação [2], como representado na figura 17.

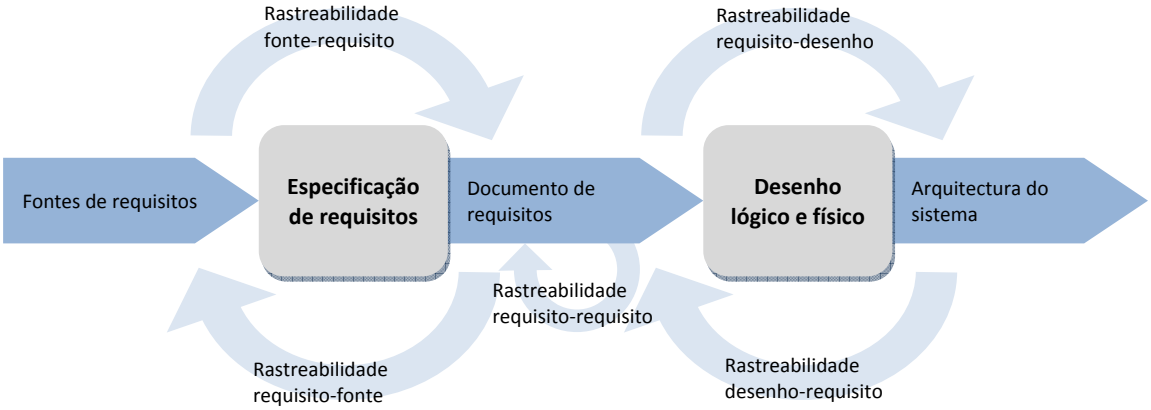

Figura 17 – Tipos de rastreabilidade no ciclo de desenvolvimento de software

- Rastreabilidade fonte-requisito – Informação que liga cada requisito às fontes do qual foi descoberto. Esta informação é incluída directamente na descrição de cada requisito e permite explicar quais as razões que levaram à sua inclusão.
- Rastreabilidade requisito-desenho – Informação que liga um componente aos requisitos que implementa e permite explicar quais as razões que levaram à sua criação. Esta informação deve ser incluída na documentação produzida durante a concepção do desenho lógico e físico do sistema.
- Rastreabilidade desenho-requisito – Informação que liga cada requisito aos componentes que o implementam. Como os detalhes do desenho e implementação não são directamente do âmbito do documento de especificação de requisitos, esta informação deve ser incluída como apêndice quando estiver disponível. Permite avaliar o impacto de uma alteração no desenho e implementação.

- Rastreabilidade requisito-fonte – Informação que liga uma fonte de requisitos aos requisitos que dela foram descobertos e que a referenciam no documento de especificação. Esta informação pode ser derivada do documento de especificação e justifica a importância de cada fonte. Permite avaliar o impacto de uma alteração na informação das fontes nos requisitos que dela foram descobertos.
- Rastreabilidade requisito-requisito – Informação que liga os requisitos documentados entre si sob a forma de dependências. Correspondem a associações bidireccionais em que se indica para cada requisito quais são aqueles dos quais ele depende.

À medida que o ciclo de desenvolvimento de software progride, as versões do documento de especificação de requisitos vão sendo enriquecidas com informação de rastreabilidade fonte-requisito, desenho-requisito e requisito-requisito.

Tabela 4 – Factores de decisão para políticas de rastreabilidade

Factor de decisão	Descrição
Número de requisitos	Quanto maior for o número de requisitos, maior será a necessidade de gerir a informação de rastreabilidade. Se o número tornar essa gestão impraticável, será preferível isolar a informação de rastreabilidade crítica a manter.
Tempo de vida útil do sistema	Só se justifica um grande controlo de rastreabilidade se o sistema tiver um tempo de vida útil longo, caso contrário, não se retiram os benefícios que justifiquem o esforço de a manter.
Nível de maturidade da organização	Organizações nos níveis mais baixos de maturidade devem focar-se somente em manter informação de rastreabilidade requisito-requisito. Organizações em níveis mais elevados, poderão ter políticas mais completas.
Dimensão da equipa	Se a equipa for pequena, a necessidade de estruturação e formalismo é menor do que para equipas grandes (mais de vinte elementos). Se as equipas não trabalharem no mesmo local é necessário trocarem entre si informação de rastreabilidade, sob risco de perderem controlo das alterações e impacto das mesmas.
Tipo de sistema	Os sistemas considerados críticos, de controlo em tempo-real necessitam mais de políticas mais completas do que sistemas não-críticos.
Requisitos específicos do cliente	Alguns clientes podem pretender que lhes seja entregue a informação de rastreabilidade. Nesse caso, é necessário negociar com eles que informação deve ser incluída.

A tarefa de recolha, análise e manutenção de informação de rastreabilidade é complexa e demorada. É necessário garantir que está correcta e actualizada quando for necessário consultá-la. Sempre que possível, deve ser gerida com uma ferramenta CASE e não manualmente. É necessário que a organização determine o tipo e quantidade de informação de rastreabilidade que pretende controlar, definindo assim uma política de rastreabilidade. A tabela 4 discrimina os factores de decisão que pode influenciar que forma assumir essa política.

2.5.3 Controlo de estados de requisitos

O estado de um requisito traduz a situação em que este se encontra no ciclo de desenvolvimento de software. O controlo de estados de requisitos é a actividade que gere as transições dos seus estados.

O registo dos estados dos requisitos é uma actividade essencial na sua gestão. Por um lado, permite obter um ponto de situação do desenvolvimento, isolando facilmente os requisitos aprovados dos que foram eliminados e rejeitados. Por outro, apoia o controlo de versões, já que a implementação de cada requisito fica associada a uma versão do produto. Serve também de base à geração automática de uma determinada versão do documento de especificação de requisitos. Embora cada organização possa adoptar a sua própria nomenclatura, Wiegers [6] sugere os estados da tabela 5.

Tabela 5 – Estados dos requisitos

Estado	Descrição
Proposto	O requisito foi identificado de uma fonte de requisitos ou de uma fonte de mudanças.
Aprovado	O requisito foi analisado e incluído nos requisitos acordados. Foi alocado a uma versão específica do produto a disponibilizar. Os principais interessados concordaram em incorporar o requisito e a equipa de desenvolvimento em desenvolve-lo.
Implementado	O código que implementa o requisito foi desenhado e codificado. Foram executados testes unitários ao código respectivo.
Verificado	Foi confirmado o correcto funcionamento do requisito implementado no contexto do produto através de testes de aceitação.
Eliminado	Um requisito que tinha sido aprovado foi eliminado da lista de requisitos acordados. É incluída uma explicação dos motivos que levaram a esta decisão e por quem foi tomada.
Rejeitado	O requisito foi proposto mas não planeado para implementação em qualquer versão futura. É incluída uma explicação dos motivos que levaram a esta decisão e por quem foi tomada.

Interpretando a informação na tabela 5 é possível estabelecer as regras de transição de estados. A figura 18 apresenta, através de um diagrama de estados, as transições de estado que um requisito pode ter no seu ciclo de vida.

Um histórico de requisitos pode ser mantido através de uma tabela que inclua para cada linha o estado do requisito, a data em que transitou para esse estado, as causas que levaram a essa transição e a pessoa responsável por essa decisão. As ferramentas de apoio à gestão de requisitos permitem registar os seus estados e gerar automaticamente o histórico a partir dessa informação.

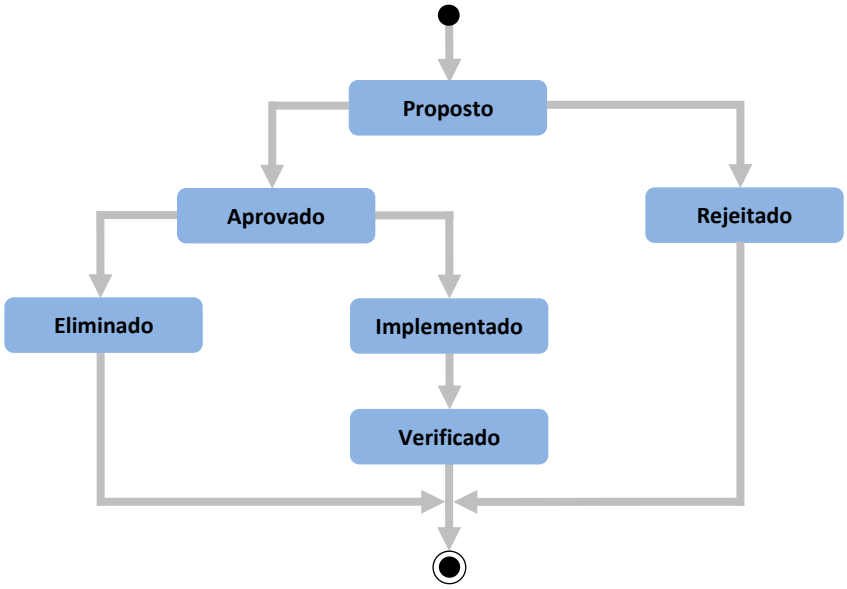

Figura 18 – Diagrama de estados de um requisito

2.5.4 Controlo de versões

O controlo de versões é a actividade final do processo de gestão de requisitos e aquele que precede a disponibilização de uma versão do documento de especificação. Destina-se a determinar quais são as alterações solicitadas que vão ser implementadas na próxima versão do produto. Essa decisão envolve a reunião das equipas de desenho e implementação que terão que ser consultadas para fornecerem uma estimativa do tempo que necessitarão para as incorporar no sistema. Dessa reunião, pode ser decidido manter alguns requisitos no estado aprovado sem os implementar, atendendo a motivos como:

- Prioridade – Algumas alterações, principalmente as que se devem a problemas inesperados, são de implementação mais urgente que outras;
- Complexidade – Existem alterações cuja implementação é mais difícil que outras.

Os requisitos que devem ser implementados e verificados farão parte da próxima versão do produto. A informação de rastreabilidade de requisito-desenho permite associar os componentes de software que têm de ser alterados para dar resposta a alterações nos requisitos. O conjunto de componentes de software alterados consubstancia as diferenças entre versões do produto.

À semelhança das outras actividades do processo de gestão de requisitos, o controlo de versões deve ser apoiado por uma ferramenta CASE. Esta ferramenta define um esquema de identificação de versões e utiliza a informação de rastreabilidade para associar versões do produto às alterações a requisitos. Quando se decide quais são os requisitos que vão constar da versão actual, gera-se uma nova versão do documento de requisito e este é entregue à equipa de desenho e posteriormente à de implementação.

2.6 Conclusões

Os requisitos são descrições do comportamento de um sistema. A sua principal classificação é em requisitos funcionais e não-funcionais. Os requisitos funcionais incidem essencialmente em informação de domínio de aplicação e descrições de casos de uso enquanto os requisitos não-funcionais incidem principalmente em restrições ao processo de desenvolvimento.

O processo de engenharia de requisitos destina-se a desenvolver novos requisitos ou gerir alterações aos já existentes, no âmbito da etapa de especificação de requisitos do ciclo de desenvolvimento de software. Engloba os sub-processos de desenvolvimento de requisitos e gestão de requisitos.

O processo de desenvolvimento de requisitos é um conjunto estruturado de actividades relacionadas com a descoberta, análise, documentação e validação dos requisitos de um sistema. O objectivo desta etapa é produzir um documento – o documento de especificação de requisitos – que permite estabelecer uma base de acordo entre os interessados do sistema, fornecer documentação ao projecto para as equipas de desenho, implementação e testes, promover a compreensão do negócio e apoiar a gestão de projectos.

Capítulo 3

A metodologia proposta

3.1 Introdução

Neste capítulo descreve-se a metodologia de desenvolvimento de requisitos proposta. A metodologia proposta baseia-se nas etapas genéricas da metodologia de desenvolvimento de requisitos de Kotonya-Sommerville [2], porém, não utiliza todas as suas técnicas. Incorpora também ideias e mecanismos de outras metodologias com propósitos específicos como a de Eriksson-Penker [13] para modelação de negócio e a de modelação com casos de uso. Representa um esforço de ajuste do processo genérico de desenvolvimento de requisitos a projectos que aderem aos pressupostos apresentados na secção seguinte.

A metodologia estrutura-se em três grandes etapas apresentadas ao longo deste capítulo. Recorrendo a esquemas representa-se o fluxo de actividades, os artefactos utilizados, os artefactos produzidos e respectivos intervenientes. Cada etapa é constituída por um conjunto de actividades que são descritas de forma detalhada.

3.2 Pressupostos

A metodologia desenvolvida é fortemente estruturada e assenta em três pressupostos. Afirma-se que é fortemente estruturada porque se apresenta como um conjunto de etapas com actividades que têm uma ordem sequencial de realização pré-definida.

Os pressupostos para a aplicação da metodologia descrita são os seguintes:

1. Utilização, para a especificação, de sistemas de informação de dimensão média que suportem processos com uma forte componente administrativa e com restrições impostas por legislação;
2. Alocação de um conjunto de pessoas, com dedicação de algumas horas por semana, para a preparação e realização de actividades relacionadas com o processo de desenvolvimento de requisitos durante a sua duração;
3. Atribuição de funções de engenheiro de requisitos a uma pessoa com dedicação exclusiva ao projecto.

Embora enumerados de forma separada, na realidade, os pressupostos estão relacionados. O primeiro pressuposto sugere que a metodologia é especialmente dirigida para realizar processos de desenvolvimento de requisitos de sistemas de informação com vários módulos (dimensão média), obrigações oficiais (forte componente administrativa) e que dependam de conformidade com legislação (restrições impostas por legislação).

Devido ao manancial de regras de negócio com que este tipo de sistemas tem que ser capaz de lidar, é necessário que exista um grande envolvimento de várias pessoas na sua especificação. Dessa proliferação de regras de negócio resultam frequentemente diferentes interpretações de legislação e diferentes formas de descrever as mesmas actividades. Para garantir uma visão única e consensual é necessário envolver pessoas com diferentes perspectivas do sistema. Por esse motivo, algumas actividades da metodologia proposta só podem ser realizadas com a colaboração de várias pessoas. Por esse motivo, há um grande enfoque na realização de vários tipos de reuniões em que, para assegurar a sua produtividade, é necessário investir tempo na sua preparação.

Alinhado com a necessidade de envolver várias pessoas no processo surge o segundo pressuposto, uma vez que se reconhece que nem sempre é possível alocar um conjunto de pessoas especificamente para processos deste tipo. É essencial que seja possível atribuir a essas pessoas algumas horas de trabalho semanal especificamente para o processo de desenvolvimento de requisitos. Caso não o seja, é preferível adoptar uma metodologia mais assente em trabalho individual. Por exemplo, utilizando técnicas de análise de sistemas existentes baseadas em engenharia reversa, caso existam sistemas legados.

O último pressuposto prevê que exista pelo menos uma pessoa com dedicação exclusiva ao processo. Adopta-se a designação de "engenheiro de requisitos" indicada na metodologia de Kotonya-Sommerville, embora outras, como "analista", sejam também admissíveis. Trata-se da pessoa que controla todo o processo e decide quando se reúnem as condições para ser realizada cada actividade. É também da sua responsabilidade a redacção da maior parte dos documentos intermédios ou definitivos.

3.3 Perspectiva geral

Na figura 19 representa-se uma perspectiva geral da metodologia de desenvolvimento de requisitos proposta, onde é possível identificar três etapas:

1. Identificação de fontes de requisitos e planeamento;
2. Análise de requisitos por áreas funcionais;
3. Especificação e documentação de requisitos.

A conclusão de cada etapa implica a produção de um ou mais artefactos que são disponibilizados ao exterior: um documento de estudo prévio, vários documentos de especificação base e um documento de especificação de requisitos. Também representados na figura estão os vários intervenientes e outros tipos de artefactos. À semelhança de outras metodologias, cada etapa é composta por actividades cuja realização pode ser exclusivamente da responsabilidade do engenheiro de requisitos atribuído ao projecto ou da responsabilidade conjunta de outros intervenientes.

A primeira etapa, além de produzir um documento de estudo prévio, fornece três elementos essenciais para a realização da etapa seguinte:

- A equipa de especificação, constituída para apoiar o engenheiro de requisitos na análise por áreas funcionais;
- Um plano de reuniões, que inclui a decomposição do sistema em partes;
- As fontes de requisitos já identificadas e utilizadas para produzir o estudo prévio. Estas serão novamente consultadas e analisadas na etapa seguinte.

Partindo do plano de reuniões realiza-se a etapa de análise de requisitos de cada parte. A equipa de especificação realiza um conjunto de reuniões com o propósito de examinar detalhadamente cada parte do sistema. O engenheiro de requisitos, baseado nessa análise e complementando-a com outras técnicas, produz documentos de especificação base referentes a cada parte.

A partir dos documentos de especificação base das várias partes do sistema, e integrando informação no documento de estudo prévio, inicia-se a terceira etapa para especificação de requisitos do sistema. Esta etapa abrange a especificação de requisitos de cada parte do sistema e a consolidação dessa informação num único documento de especificação de requisitos. A consolidação pode ser realizada no fim do processo ou de forma incremental, a seguir à especificação de requisitos de cada

parte. Na perspectiva genérica representada na figura 19 omitiram-se os detalhes relacionados com as actividades intermédias e artefactos internos que são produzidos em cada etapa.

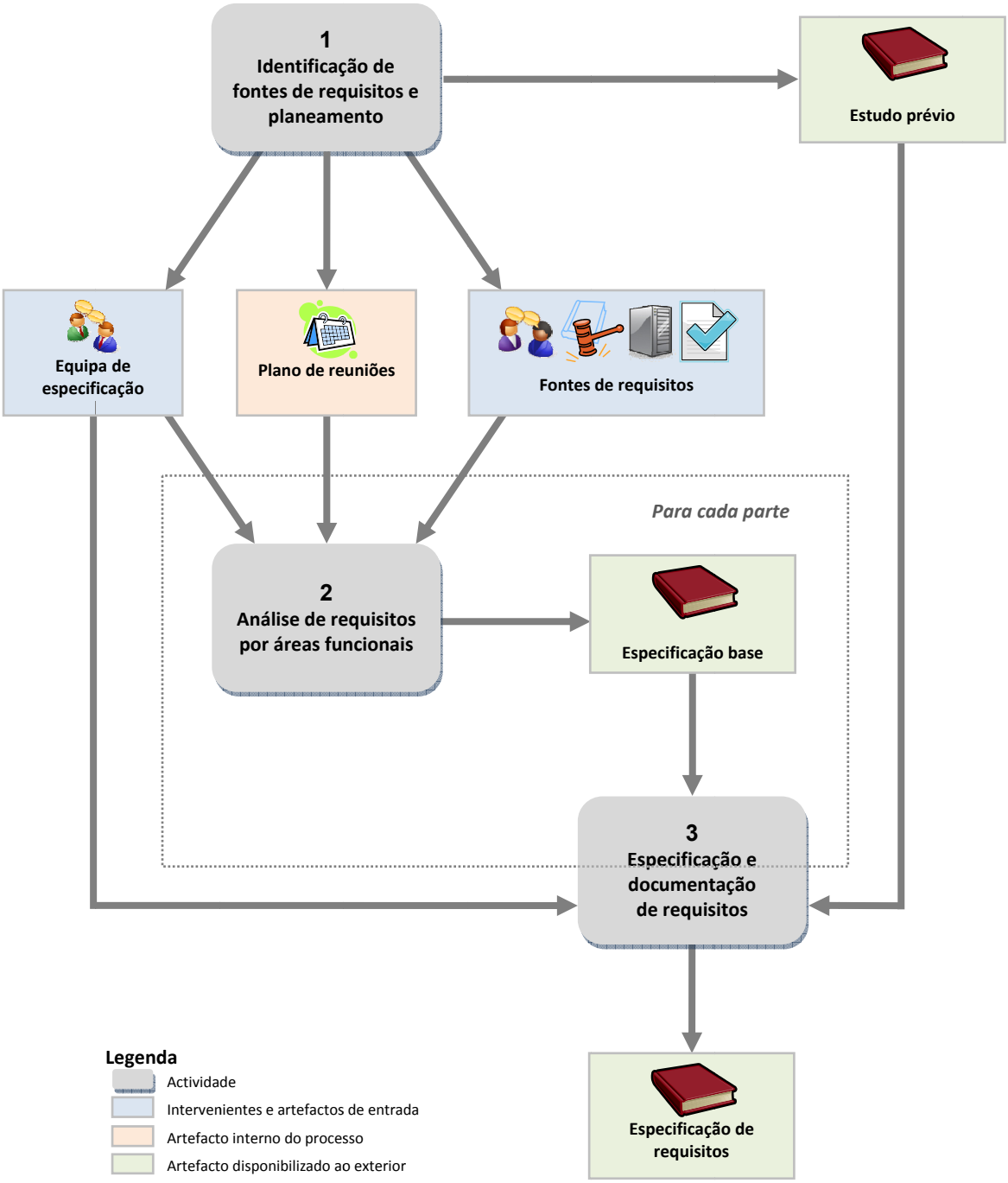

Figura 19 – Perspectiva geral da metodologia proposta

Seguidamente, cada etapa é descrita em detalhe, definindo-se claramente as técnicas utilizadas e o formato de cada um dos artefactos produzidos.

3.4 Etapa de identificação de fontes de requisitos e planeamento

A metodologia de desenvolvimento de requisitos proposta inicia-se com a etapa de identificação de fontes de requisitos e planeamento. Este etapa destina-se a lançar as bases necessárias para ser possível efectuar-se a análise de requisitos por áreas funcionais. A figura 20 representa em detalhe esta etapa com as actividades realizadas e os artefactos e intervenientes envolvidos em cada uma.

3.4.1 Identificação das fontes de requisitos

A etapa inicia-se com a actividade de identificação das fontes de requisitos (actividade 1.1). Consideram-se fontes de requisitos todos os elementos que podem conter informação útil para descobrir os requisitos do sistema a desenvolver. No contexto desta metodologia e atendendo aos seus pressupostos, as fontes de requisitos são as seguintes:

- Interessados (*stakeholders*) no sistema – É o conjunto de todas as pessoas que estão envolvidas no sistema, tendo, por diferentes motivos, interesse no seu desenvolvimento. Normalmente, neste conjunto, incluem-se os donos do negócio, os potenciais utilizadores e as equipas de desenvolvimento. Os interessados possuem conhecimento importante que deve ser utilizado no processo.
- Legislação, regulamentos e documentação do domínio – É o conjunto de todos os documentos que, de diferentes formas, definem terminologia e descrevem regras de negócio. A legislação é um conjunto de diplomas emitidos pelo Governo que condiciona a forma de operacionalizar processos. Os regulamentos são documentos internos da organização que definem os procedimentos e orientam a realização de processos. A documentação de domínio é o conjunto de todos os documentos, livros e publicações das quais é possível extrair conhecimento genérico sobre a área de negócio em que o sistema se enquadra.
- Sistemas existentes – É o conjunto de todos os sistemas informáticos existentes na organização, que já apoiam em parte ou na totalidade processos de negócio, que o novo sistema se destina a suportar. Destes sistemas, os que forem descontinuados e substituídos pelo novo passarão a ter o estatuto de "sistema legado".
- Objectivos gerais do sistema – É o conjunto dos principais motivos que fundamentam o desenvolvimento do novo sistema. São normalmente indicados directamente pelos donos do negócio de forma genérica.

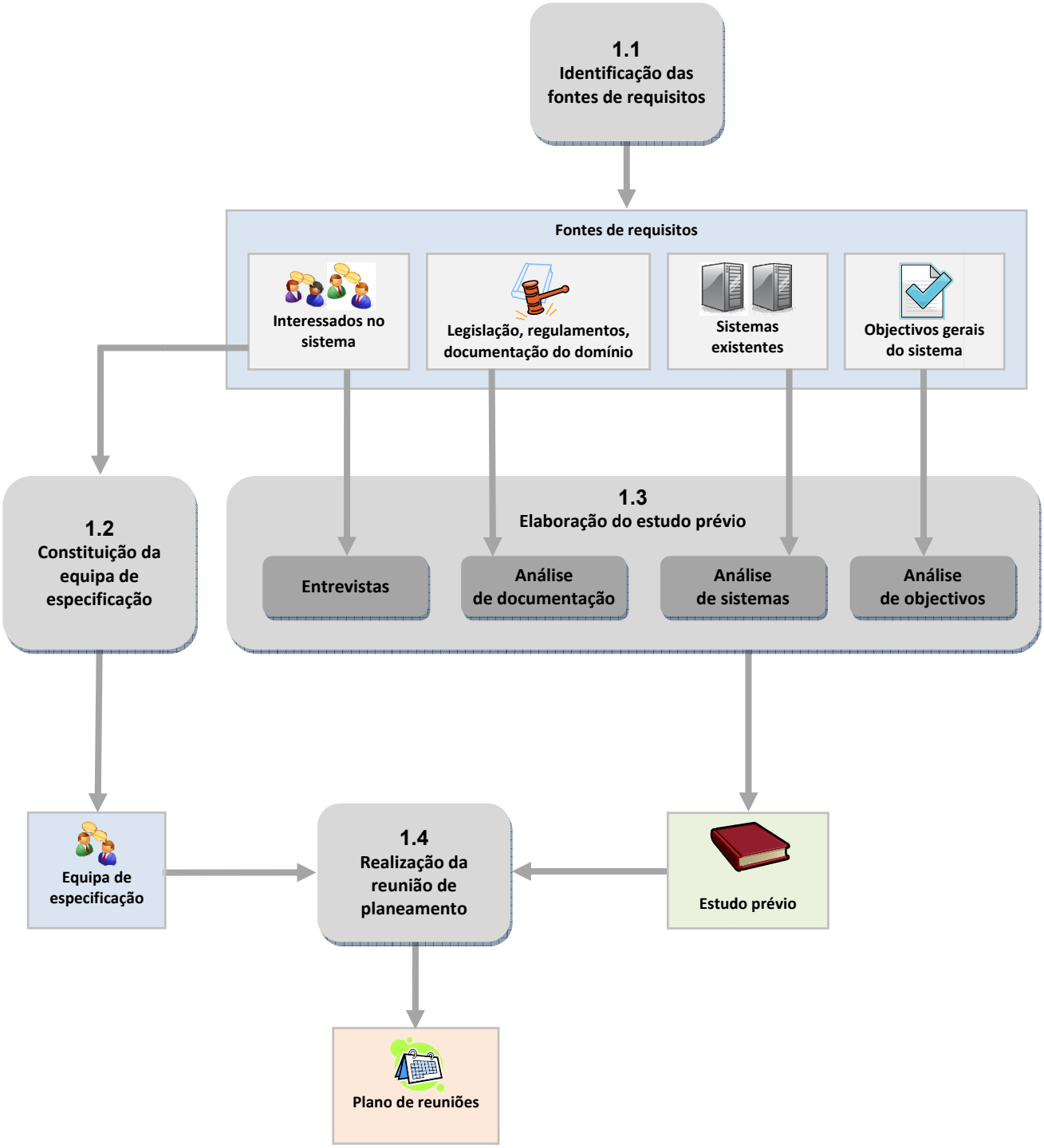

Figura 20 – Etapa de identificação de fontes de requisitos e planeamento

A actividade de identificação de fontes de requisitos destina-se exclusivamente a reconhecer quais são as fontes que o engenheiro de requisitos deve considerar para identificar os requisitos do sistema.

3.4.2 Constituição da equipa de especificação

Do conjunto dos interessados do sistema é constituída uma equipa de especificação (actividade 1.2). Esta equipa deve incluir pessoas com conhecimentos do processo do negócio a suportar e com boa capacidade de análise. Preferencialmente deve incluir um ou dois representantes dos utilizadores do sistema, um ou dois elementos da equipa de desenvolvimento e um moderador. Eventualmente podem ser incluídas outras pessoas, não necessariamente interessadas no sistema, desde que possuam as qualidades enunciadas e para as quais seja reconhecido claramente que há benefícios no seu envolvimento. O próprio engenheiro de requisitos é incluído na equipa.

3.4.3 Elaboração do estudo prévio

Em paralelo com a constituição da equipa de especificação, o engenheiro de requisitos inicia a elaboração do estudo prévio (actividade 1.3). Este documento destina-se a apresentar um resumo do conhecimento base extraído a partir das fontes de requisitos para fornecer os elementos mínimos necessários para orientar a etapa seguinte de análise de requisitos por cada área funcional.

Nesta metodologia, para extrair informação de cada tipo de fontes de requisitos, aplicam-se as seguintes técnicas:

- Entrevistas – Destinam-se a extrair informação dos interessados do sistema, sendo previamente preparadas pelo engenheiro de requisitos. Por esse motivo, é preferível que seja a última técnica a aplicar depois de reunido o conhecimento obtido pelas outras técnicas.
- Análise de documentação – Destina-se a organizar a legislação, regulamentos e documentação de domínio previamente identificados. Implica efectuar uma operação de triagem para identificar quais os documentos que apresentam interesse real para a especificação. Simultaneamente classificam-se os documentos de forma a formar grupos por áreas de negócio, processos ou outros critérios. Os grupos criados servem de base para o particionamento do sistema que será necessário efectuar para a etapa de análise de requisitos.
- Análise de sistemas – Destina-se a extrair informação a partir dos sistemas existentes na organização, em particular, sobre a forma como apoiam o processo de negócio envolvido no novo sistema. Esta análise resulta de actividades de observação de interacções com os sistemas pelos utilizadores habituais ou realizadas pelo próprio engenheiro de requisitos. Inclui também a consulta de documentação dos sistemas, sob a forma de manuais de utilizador e, caso existam, documentos de especificação de

requisitos realizados no seu desenvolvimento. Neste âmbito deve obter-se informação sobre a arquitectura física e lógica dos sistemas, uma vez que a última também fornece bases para o particionamento do novo sistema.

- Análise de objectivos – Destina-se a organizar os objectivos gerais do sistema e enquadrá-los nos objectivos da organização. Implica relacionar as informações obtidas de outras fontes de requisitos, de forma a compreender o impacto que o novo sistema terá na organização e, se for o caso, na substituição de outros sistemas.

Note-se que, sempre que necessário, o engenheiro de requisitos utilizará a informação extraída de uma técnica para iniciar ou completar o processo de extracção de informação de outra.

Parte da informação recolhida é reunida para produzir um documento de estudo prévio. Este documento tem a estrutura representada na tabela 6.

Tabela 6 – Estrutura do documento de estudo prévio

Capítulo	Descrição
1. Introdução	Capítulo para apresentação do propósito do documento e da forma como está organizado.
2. Objectivos	Capítulo para identificação e breve explicação dos objectivos do sistema, enquadrando-os com os da organização.
3. Estrutura da organização e processos	Capítulo para apresentar a descrição da estrutura da organização e uma perspectiva geral dos processos que o sistema se destina a suportar.
4. Apoio de tecnologias de informação	Capítulo que apresenta uma breve panorâmica da utilização de tecnologias de informação para apoio do processo. Inclui uma descrição de cada sistema existente. Pode ser complementada com um pequeno estudo de outros sistemas concorrentes e das principais funcionalidades encontradas em cada.

Recomenda-se que o documento inclua uma secção no início com uma tabela de histórico de versões do documento.

3.4.4 Realização da reunião de planeamento

Com a equipa de especificação constituída e o estudo prévio elaborado reúnem-se as condições para ser realizada a reunião de planeamento (actividade 1.4). O seu propósito é orientar a etapa seguinte da metodologia, em que são analisadas em detalhe várias áreas de negócio que corresponderão a partes do sistema.

O artefacto que resulta da reunião de planeamento é o plano de reuniões. Este inclui a previsão de reuniões de especificação e validação a realizar para cada parte do sistema.

A reunião de planeamento deve ser conduzida de forma a discutir três tópicos:

- Definição das funções da equipa – Após a apresentação da equipa e distribuição de contactos, identificam-se claramente as funções de cada membro:
  - Moderador – Orientar a discussão e assegurar que todas as partes são ouvidas;
  - Consultor – Analisar e discutir os tópicos elencados para cada reunião de acordo com a sua perspectiva;
  - Engenheiro de requisitos – Observar e registar as ideias e conclusões das reuniões com o propósito de identificar requisitos.
- Particionamento do sistema – A equipa, baseando-se na informação do estudo prévio sobre os macro-processos do processo principal a suportar e da arquitectura lógica de outros sistemas já existentes, propõe uma forma de particionar o sistema. O objectivo do particionamento é decompor o sistema de forma a permitir uma análise autónoma de cada parte. Não se utiliza intencionalmente a designação de "módulo". Tal deve-se ao facto de que cada parte nem sempre corresponder a um futuro módulo do sistema. Por outro lado, várias partes poderão vir a ser incorporadas num único módulo. Não sendo o objectivo da especificação de requisitos definir a arquitectura lógica do novo sistema, não é adequado impor uma estruturação em módulos. Tal deve ser remetido para a etapa do ciclo de desenvolvimento de software dedicada ao desenho lógico do sistema. A designação "parte" é mais genérica e não sugere a imposição de uma estrutura lógica para o novo sistema.
- Planeamento – A equipa de especificação propõe um plano para as reuniões de especificação e validação a realizar. Cada reunião será dedicada a uma parte do sistema, podendo haver mais do que uma reunião para cada parte. Para além da sua calendarização, é necessário que as reuniões sejam preparadas previamente. Essa responsabilidade é repartida pelos vários membros da equipa. A preparação de cada reunião é atribuída a um membro atendendo ao conhecimento que possua sobre os tópicos da parte a analisar. É previsível que, à medida que o processo de especificação progrida, seja necessário efectuar periodicamente ajustes ao planeamento para que este reflicta a situação actual e simultaneamente uma previsão realista das reuniões futuras.

O plano de reuniões apresenta-se sob a forma de uma tabela em que se indicam a data de realização de cada reunião, parte a que se refere, respectivo responsável. À medida que as reuniões vão sendo realizadas, pode completar-se também com os tópicos que foram discutidos em cada uma.

3.5 Etapa de análise de requisitos por áreas funcionais

Com base no particionamento indicado no plano de reuniões, cada parte do sistema é alvo das actividades da etapa de análise de requisitos por áreas funcionais. O artefacto que resulta desta etapa é um documento de especificação base que reúne o conhecimento considerado essencial sobre a parte analisada e que será utilizado na etapa seguinte para descrever os requisitos do sistema. Haverá um documento de especificação base para cada parte. A figura 21 representa em detalhe esta etapa com as actividades realizadas, os artefactos utilizados e produzidos e os intervenientes envolvidos em cada uma.

3.5.1 Preparação de reuniões de especificação

O plano de reuniões indica quais são as reuniões de especificação a realizar, em que data e que membro da equipa de especificação as deve preparar. Dessa forma, esse membro inicia a preparação das reuniões (actividade 2.1) que lhe foram atribuídas. O esforço necessário nessa preparação deve focalizar-se em:

- Identificar as principais actividades realizadas na área funcional analisada;
- Apresentar a forma como os sistemas actuais suportam essas actividades;
- Distinguir e elencar os tópicos que parecem consensuais dos que devem ser levados a discussão.

No final é necessário que surja uma lista de tópicos para discussão. Nesta actividade não é importante que essa lista siga nenhum formalismo específico. A sua intenção é capturar ideias de forma a orientar as reuniões e torná-las o mais produtivas possível.

Para preparar as reuniões de especificação o responsável terá necessariamente que utilizar as fontes de requisitos previamente identificadas pelo engenheiro de requisitos. Pode, eventualmente, identificar novas fontes relacionadas directamente com a área funcional analisada e, nesse caso, alertar o engenheiro de requisitos que deve incluir essas fontes nas que já identificou.

3.5.2 Realização de reuniões de especificação

As reuniões de especificação (actividade 2.2) realizam-se para discutir tópicos sobre uma parte do sistema, normalmente afecta a uma área funcional. A primeira reunião no contexto de uma parte deve ser iniciada com uma panorâmica geral de todos os tópicos que serão discutidos. As reuniões seguintes devem iniciar-se com uma revisão geral sobre os tópicos que já foram discutidos e os que ainda resta abordar. A frequência da realização das reuniões é estabelecida no plano de reuniões. A sua duração é variável, contudo, não deve exceder as três horas de duração sob risco da diminuição da produtividade dos membros da equipa.

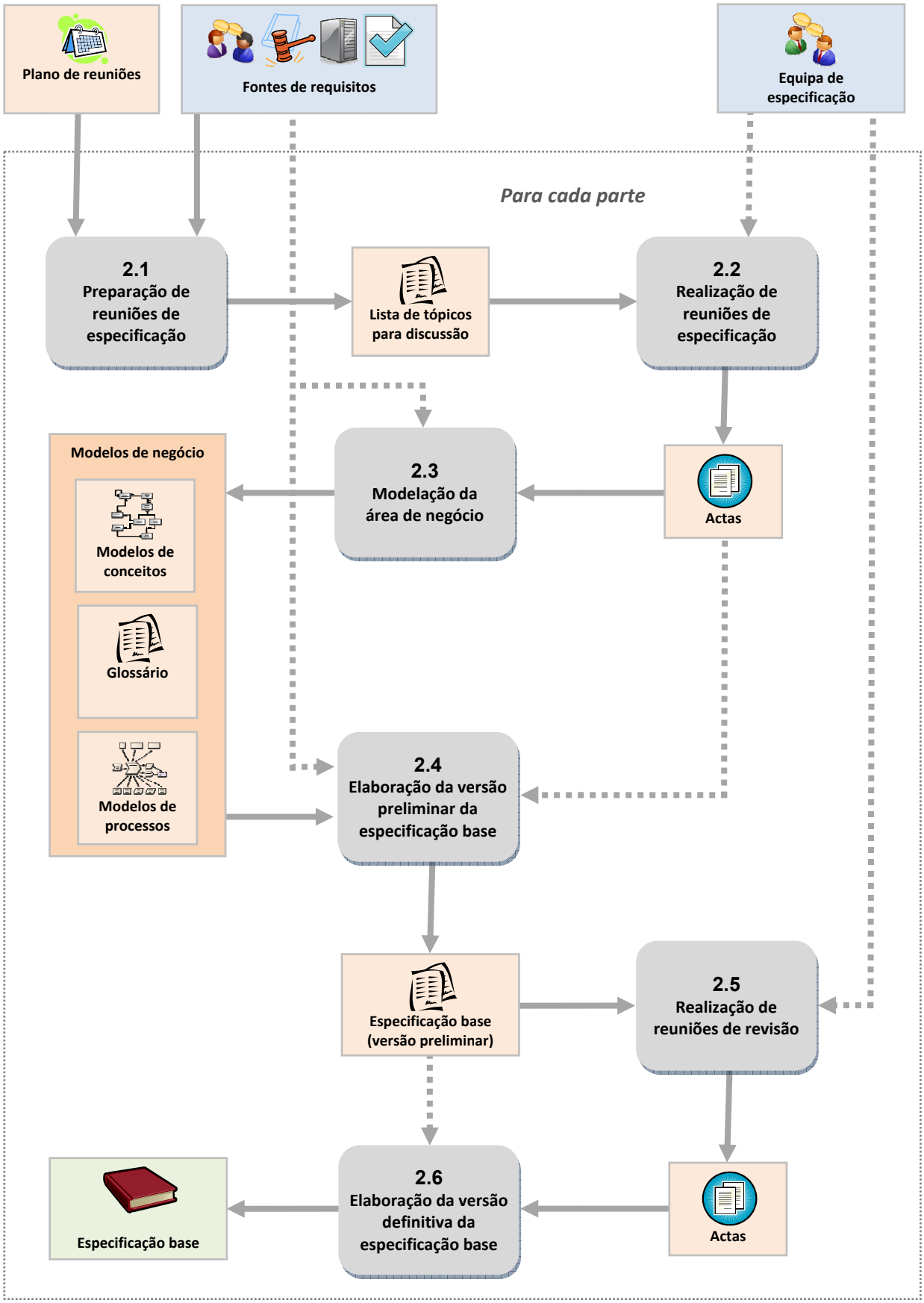

Figura 21 – Etapa de análise de requisitos por áreas funcionais

Verifica-se que é preferível que ocorram várias reuniões sobre o mesmo tema do que uma só demasiado extensa. O membro responsável pela preparação apoia o moderador na orientação da discussão, apresentando cada tópico e colocando questões ou duvidas aos outros intervenientes, lançando dessa forma a discussão. Durante a reunião, qualquer membro da equipa pode intervir e lançar também novos tópicos de discussão, desde que tal não implique grande alteração no plano estabelecido para a mesma e não provoque atrasos. É da responsabilidade do moderador assegurar que a reunião cumpre os seus propósitos e se cinge ao contexto da parte do sistema analisada.

À medida que a reunião vai decorrendo, as funções do engenheiro de requisitos são o registo de informação sobre:

- Os assuntos considerados consensuais;
- As conclusões obtidas sobre os assuntos menos consensuais e que foram alvo de discussão;
- Novas ideias sobre tópicos de discussão e novos tópicos não previstos.

Para cada reunião haverá uma acta que consolida a informação registada. As actas podem ser elaboradas pelo moderador da reunião ou pelo engenheiro de requisitos. Devem conter os elementos indicados na tabela 7.

Tabela 7 – Estrutura de actas de reuniões

Elemento	Descrição
Número	Indicação do número da reunião.
Assunto	Indicação da parte do sistema discutida.
Data	Data da realização da reunião.
Presentes	Membros da equipa de especificação presentes. Idealmente toda a equipa de especificação deve estar presente. O moderador e o engenheiro de requisitos têm presença obrigatória, os outros membros podem estar ausentes. Podem ser convidadas outras pessoas cuja contribuição tenha particular interesse para discutir os tópicos da reunião.
Ordem de trabalhos	Resumo dos tópicos que se pretende discutir e outros que, embora não elencados inicialmente, tenham sido abordados na reunião.
Conteúdo	Narrativa estruturada por itens das principais ideias registadas (conclusões, dúvidas, reflexões, etc.) apresentando-as pela ordem com que foram discutidas.

3.5.3 Modelação de áreas de negócio

A modelação de negócio é uma actividade complexa e destina-se a especificar as necessidades de um negócio e como este funciona. No contexto da metodologia apresentada, a modelação é realizada por cada área de negócio (actividade 2.3) sempre que a parte do sistema esteja directamente relacionada com um sub-processo.

Os modelos de negócio têm vindo a ser incorporados como instrumento de análise de requisitos e são determinantes para identificar as partes do negócio que devem ser apoiadas por sistemas de software. É frequente encarar-se a percepção do negócio como parte da descrição do problema e a especificação de requisitos como parte da descrição da solução. Conceitos do negócio como objectivos, processos, recursos, regras, intervenientes e eventos são essenciais para a descoberta dos requisitos funcionais e não-funcionais do sistema de informação de suporte ao negócio.

Tabela 8 – Objectivos da modelação de negócio

Motivo	Descrição
Criação de sistemas de informação adequados ao suporte do negócio	As descrições do negócio são usadas para identificar as partes em que é necessário o suporte pelo sistema de informação. Os modelos são também usados como base para especificar os requisitos desses sistemas.
Melhoria das estruturas de negócio actuais e da sua operacionalização	Os modelos ajudam a identificar alterações ao negócio actual que é necessário implementar para o melhorar.
Inovação da estrutura do negócio	O modelo torna-se a base para o plano de acção. Inovação sugere que alterações radicais (em vez de alterações incrementais) foram efectuadas aos processos do negócio.
Exploração de novos conceitos para o negócio	O modelo de desenvolvimento torna-se um esboço de um possível desenvolvimento no negócio. O modelo pode ter uma ideia nova, inspirada pela modelação de outros negócios.
Identificação de oportunidades de subcontratação	As partes do negócio que não são consideradas "principais" são remetidas para fornecedores exteriores à organização. Os modelos são usados como especificação para os fornecedores.
Servir de base para certificação de qualidade	Os modelos constituem um vocabulário universalmente aceite que promove a certificação de qualidade por entidades acreditadas.
Servir de base para manuais de procedimentos	Os modelos representam de forma concisa a complexa realidade da organização facilitando a disseminação de informação.

A tabela 8 resume os principais objectivos que, segundo Eriksson e Penker [13], levam a que uma metodologia de desenvolvimento de requisitos deva incluir a modelação de negócios. Na sua perspectiva, à medida que aumenta a infra-estrutura do sistema de software, aumenta também o potencial dos sistemas desenvolvidos ficarem mais orientados ao negócio. Por consequência, as equipas de desenvolvimento podem concentrar-se mais nas funcionalidades que suportam o negócio em vez de resolver incompatibilidades técnicas ou outros problemas.

A tabela 9 resume algumas das metodologias de modelação de processos, realçando os aspectos que as particularizam. Segundo a Sparx Systems [14], fabricante do IDE de produção de modelos em UML *Enterprise Architect*, as duas metodologias cujas notações têm maior aceitação são a BPM e a Eriksson-Penker. Considera que a metodologia BPM foi a que ganhou maior popularidade e está rapidamente a tornar-se um novo padrão para modelizar e desenhar processos de negócio.

Tabela 9 – Algumas metodologias de modelação de processos

Metodologia	Notação	Aspectos particulares
Enterprise Knowledge Development (EKD) [15]	Própria	Modelos de negócio completos.
Marshall [16]	UML	Modelação de objectivos, processos, entidades e organização.
Eriksson-Penker [13]	UML	Modelação do negócio com extensões à UML.
Rational Unified Process (RUP) [1]	UML	Modelação de negócio com casos de uso e objectivos de negócio.
Business Modeling Method (BMM) [17]	UML	Inclusão das tecnologias a utilizar nos modelos.
Business Process Management (BPM) [18]	BPMN	Modelos executáveis em linguagens como BPML ou BPEL.

A metodologia de Eriksson-Penker, por outro lado, fornece meios únicos e eficazes para visualizar e comunicar processos de negócio e o fluxo de informação numa organização. O facto de utilizar a mesma notação para o negócio e para o software reduz a barreira semântica entre os dois tipos de modelos e, simultaneamente, aumenta a rastreabilidade dos requisitos do sistema. Por esse motivo, e porque não é objectivo da metodologia de desenvolvimento de requisitos proposta produzir modelos executáveis, adoptou-se a de Eriksson-Penker para a actividade de modelação de áreas de negócio. No anexo A discutem-se os principais conceitos de negócio e as diferentes perspectivas propostas pela metodologia de modelação de negócio de Eriksson-Penker.

Da aplicação dessa metodologia realizam-se as sub-actividades:

- Modelação de conceitos para capturar os conceitos centrais da área de negócio. Os artefactos a produzir nesta actividade são:
  - Diagrama de conceitos para visualizar de forma integrada os conceitos e as relações entre si;
  - Obter uma definição para cada conceito e construir incrementalmente um glossário contendo essas definições.
- Modelação de processos para capturar os principais processos e actividades da área de negócio. O artefacto a produzir nesta actividade são modelos de processos que se destinam-se a:
  - Representar todos os processos, incluindo os que ainda são realizados manualmente, em diferentes níveis de detalhe;
  - Identificar claramente as actividades de cada processo que são apoiadas por tecnologias de informação e/ou por sistemas existentes;
  - Detectar actividades que podem vir a ser apoiadas pelo novo sistema.

Os modelos de negócio tornam-se assim ferramentas de análise e devem fazer parte do documento de especificação base.

3.5.4 Elaboração da versão preliminar da especificação base

Após a realização de todas as reuniões de especificação e, quando aplicável, obtidos os modelos de negócio relevantes, o engenheiro de requisitos inicia a elaboração da versão preliminar de um documento de especificação base (actividade 2.4).

A especificação base é um documento que tem o propósito de explorar o conhecimento base que se considera necessário sobre a parte do sistema analisada. O documento é elaborado utilizando e combinando a informação obtida de:

- Consulta e análise das fontes de requisitos directamente relacionadas com a parte analisada;
- Modelos de negócio elaborados;
- Actas de reuniões de especificação.

O documento deve ser preparado já com a estrutura final pretendida, mas ainda numa versão que terá que ser sujeita a uma revisão pela equipa de especificação. Por esse motivo, este artefacto designa-se versão preliminar do documento de especificação base.

Os documentos de especificação base constituem referências com informação de apoio à especificação de requisitos. Devem incluir todos os tópicos considerados de interesse para a parte do sistema analisada e também ser complementados com modelos de negócio. Na versão preliminar, representam a interpretação do engenheiro de requisitos das conclusões das reuniões de especificação, completadas com modelos de negócio.

Os documentos de especificação base devem seguir a estrutura da tabela 10.

Tabela 10 – Estrutura de um documento de especificação base

Capítulo	Descrição
1. Introdução	Resumo do documento e indicação das reuniões decorridas sobre a parte do sistema analisada.
2. Legislação e regulamentos	Indicação da legislação e regulamentos aplicáveis à parte do sistema analisada de forma estruturada.
3. Tópicos de análise	Conjunto de tópicos de discussão sobre a parte analisada. Cria-se uma subsecção para cada tópico onde são clarificados conceitos, indicadas as principais conclusões e alguns requisitos descobertos.
4. Organização de conceitos	Definições dos conceitos discutidos e apresentação de um diagrama conceptual que evidencia a sua relação.
5. Processos de negócio	Apresentação de diagramas dos principais processos relacionados com a parte do sistema analisada em diferentes níveis de detalhe e acompanhados de descrições explicativas. Trata-se de um capítulo opcional a incluir somente se tiver sido efectuada a modelação do negócio para a parte analisada.
6. Referências	Indicação das referências utilizadas para escrever o documento. Algumas delas são fontes de requisitos.

3.5.5 Realização de reuniões de revisão

Após a escrita da versão preliminar do documento de especificação base referente a uma parte do sistema, realiza-se uma ou mais reuniões de revisão (actividade 2.5). Estas têm como propósito validar essa versão do documento, para ser possível obter-se a versão definitiva. Cada membro da equipa de especificação analisa a versão preliminar do documento de especificação base e sugere alterações e correcções a ser incorporadas na versão definitiva.

O formato da reunião é semelhante ao das reuniões de especificação, contudo, com propósito diferente. Nestas reuniões é possível realizar a análise de requisitos sugerida por Kotonya e Sommerville, nomeadamente, análise de completude, dependências, sobreposições, isenção, necessidade, consistência, âmbito, viabilidade e risco. Deve também procurar-se antecipar os requisitos não-funcionais que estejam relacionados com a parte que está a ser tratada. Após discussão identificam-se os requisitos que devem ser rejeitados e quais têm que ser ajustados para resolver eventuais conflitos. O objectivo é obter acordos que permitam que seja possível elaborar a especificação base na sua versão definitiva que servirá de referência para o documento de especificação de requisitos. Da mesma forma que nas reuniões de especificação, para cada reunião de revisão é redigida uma acta para registo das decisões e conclusões obtidas.

3.5.6 Elaboração da especificação base definitiva

A elaboração da especificação base definitiva (actividade 2.6) é a última actividade da etapa de análise de requisitos por áreas funcionais. Partindo da versão preliminar elaborada previamente, o engenheiro de requisitos efectua todas as alterações ao documento necessárias para incorporar as correcções indicadas pela equipa de especificação nas reuniões de revisão.

A especificação base definitiva corresponde à versão pretendida deste documento e pode ser disponibilizada ao exterior. A versão preliminar da especificação base corresponde a um estado intermédio, um artefacto de tipo *working-draft* e não é disponibilizada ao exterior, uma vez que constitui um documento inacabado e com potenciais erros.

O principal objectivo da especificação base definitiva é fornecer informação de apoio ao documento de especificação de requisitos. Não sendo possível colocar nesse documento toda a informação, remete-se para a especificação base aquela que, não sendo considerada essencial para a especificação formal, é fundamental para que se compreenda como se chegou aos requisitos. Como sugerido na figura 22, as especificações base actuam como "pontes" entre as fontes e a especificação formal dos requisitos. Ao estabelecer essa ligação facilita as actividades de gestão de requisitos pois permite manter mais facilmente informação de rastreabilidade.

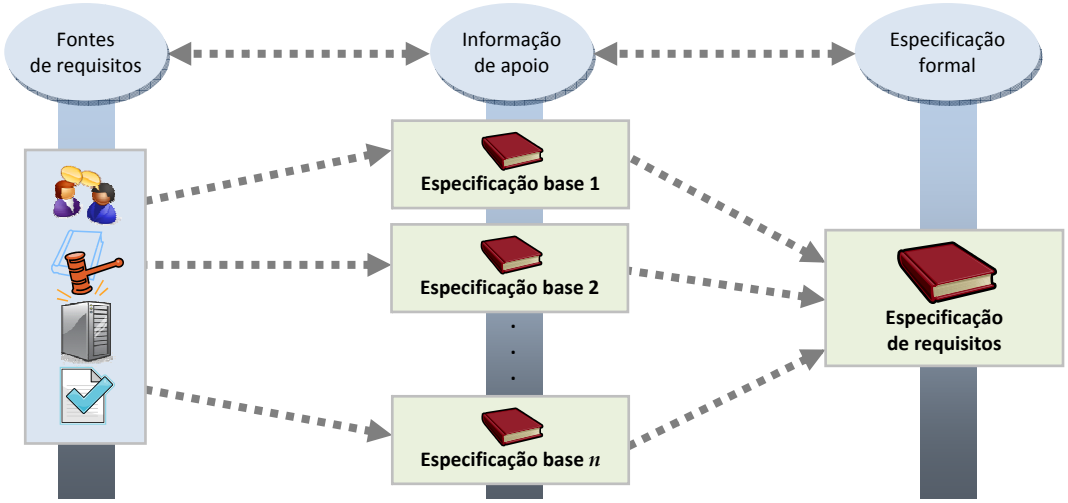

Figura 22 – O papel das especificações base

O facto de, na denominação do documento, se utilizar a palavra "definitiva" não implica que o documento não possa ser posteriormente actualizado e ter novas versões. Pretende significar que a versão produzida nesta actividade já possui o nível de qualidade suficiente para poder ser referenciada no documento de especificação de requisitos.

Em resumo, incluindo a lista de tópicos obtida na preparação das reuniões de especificação, pode considerar-se que a especificação base é um documento que possui três versões, como apresentado na tabela 11.

Tabela 11 – Versões da especificação base

Versão	Responsável por elaboração	Formato	Objectivos
Lista de tópicos	Membro responsável por preparação da parte analisada	• Não estruturado • Estilo de escrita informal	• Orientar reunião de especificação • Servir de base a acta
Preliminar	Engenheiro de requisitos	• Estruturado • Estilo de escrita formal e informal	• Interpretar as conclusões das reuniões de especificação • Obter consenso da equipa de especificação
Definitiva	Engenheiro de requisitos	• Estruturado • Estilo de escrita formal	• Fornecer informação de apoio à especificação de requisitos • Rastrear requisitos às suas fontes

3.6 Etapa de especificação e documentação de requisitos

A etapa de especificação e documentação de requisitos conclui a metodologia de desenvolvimento de requisitos proposta. O seu propósito é, utilizando os artefactos produzidos pelas etapas anteriores, obter o documento de especificação de requisitos que descreve o sistema a desenvolver. A figura 23 representa em detalhe

esta etapa com as actividades realizadas, os artefactos e os intervenientes envolvidos em cada uma. À semelhança da etapa de análise de requisitos por áreas funcionais, esta etapa possui actividades que são realizadas para cada parte do sistema, nomeadamente as modelação de casos de uso, formalização de requisitos e validação da especificação parcial. Porém, a etapa integra também uma actividade transversal de consolidação do documento de requisitos que é realizada no fim do processo.

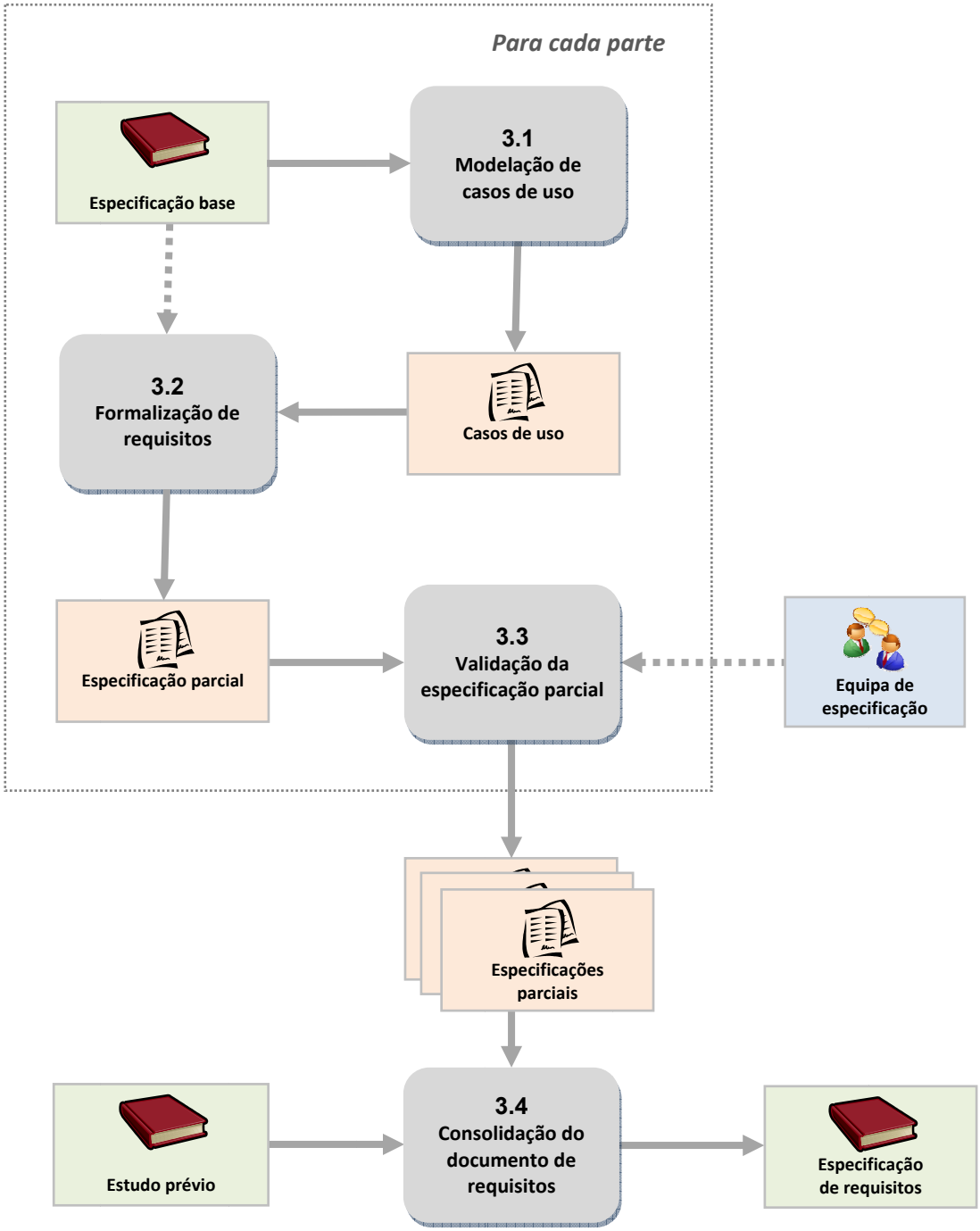

Figura 23 – Etapa de especificação e documentação de requisitos

3.6.1 Modelação de casos de uso

A etapa de especificação e documentação de requisitos inicia-se com a actividade de modelação de casos de uso (actividade 3.1). Utilizando a informação documentada da especificação base, o engenheiro de requisitos procura obter os casos de uso para o sistema a desenvolver. A modelação de software por casos de uso é uma técnica que assenta no pressuposto de que para se compreender o que um sistema deve fazer é necessário investigar como se pretende que seja utilizado e com que objectivos. Teve as suas origens nos meados dos anos 80, quando Jacobson [19] apresentou a ideia de cenários de utilização de sistemas de software.

O propósito da modelação de software por casos de uso é descrever de forma sucinta e acessível o comportamento de um sistema na perspectiva de uma "caixa preta", ou seja, apenas em termos do que é percepcionado a partir do exterior. O conjunto de casos de uso de um sistema descreve todas as formas com que este pode ser utilizado e o comportamento que apresenta em cada uma. Em consequência, clarifica o seu âmbito de actuação nos processos da organização em que se insere.

Na tabela 12 apresentam-se vários contextos, descritos por Gorman [20], em que pode ser realizada a metodologia de modelação de casos de uso de um sistema.

Tabela 12 – Motivos para modelação de software por casos de uso

Motivo	Descrição
Captura de requisitos funcionais	Os casos de uso descrevem as formas de utilizar um sistema de software para realizar funções com valor para os utilizadores, orientadas a funcionalidades. A cada caso de uso associam-se cenários de utilização do sistema.
Planeamento e estimação	Os casos de uso servem de base para o planeamento de projectos e para orientar o processo de desenvolvimento. A cada caso de uso podem ser associadas métricas que, a partir de dados históricos, auxiliem a estimar quanto tempo e esforço poderá levar à sua implementação.
Testes de sistema	Os cenários de casos de uso podem servir de base à concepção de testes. A equipa de testes instancia vários testes, com diferentes conjuntos de dados para cada cenário descrito. Pode-se eventualmente derivar mais cenários para além dos descritos (que são só os essenciais).
Documentação para o utilizador	As descrições de cenários nos casos de uso servem de base à produção de documentação para os utilizadores do sistema. É, posteriormente, completada com exemplos de interface, donde só pode ser efectuada após os requisitos estarem completos.
Simulações de negócio	As descrições de casos de uso podem ser incorporadas nas descrições do próprio negócio para que se possa simular a execução dos processos de negócio e facilmente verificar casos de uso no contexto em que serão utilizados. Permite auxiliar o enquadramento dos casos de uso nos processos de negócio.

Dos motivos apresentados na tabela 12 conclui-se que as descrições do sistema, recorrendo a casos de uso, têm como público-alvo todos os possíveis interessados no sistema. Para os clientes, as descrições dos casos de uso permitem garantir e validar que o sistema será construído proporcionando os objectivos que dele

pretendem obter. Para os gestores de projecto, oferecem uma visão suficientemente genérica do que o sistema terá de fazer para poderem eficazmente planear e monitorizar o projecto. Para a equipa de especificação, constituem mais uma forma de descrever e documentar o que o sistema irá fazer. Para a equipa de desenho e desenvolvimento, servem de ponto de partida para compreender o que sistema necessita para a sua concepção. Para as equipas de teste, facilitam a percepção do que é possível fazer-se com o sistema para poderem verificar a conformidade com a especificação de requisitos e para conceber casos de teste. Mesmo para técnicos de apoio (*helpdesk*), as descrições dos casos de uso são úteis, porque apresentam cenários típicos de utilização necessários para o apoio e formação dos utilizadores.

A modelação de casos de uso tem por base três conceitos fundamentais: actores, casos de uso e cenários de utilização. Os actores são as entidades externas que interagem com o sistema. Os casos de uso representam objectivos que um ou mais actores pretendem do sistema. Um cenário de utilização representa uma forma possível de realizar um caso de uso [21]. A metodologia de modelação de software com casos de uso é descrita no anexo B.

A especificação formal da UML [8] descreve os diagramas de casos de uso para os representar graficamente, contudo, não define nenhum formato para estruturar as suas descrições. Embora, a notação gráfica seja importante, deve sempre ser acompanhada da descrição, sob a forma de um conjunto de elementos textuais que expliquem cada caso de uso. Vários autores têm proposto elementos para organizar uma estrutura de descrição. Na tabela 13 indica-se a proposta de Bittner e Spence [22].

Tabela 13 – Elementos de estruturação de descrições de casos de uso

Elemento	Descrição
Nome	O nome do caso de uso. Cada caso de uso deve ter um nome que indique o que é obtido na sua interacção com os actores. O nome pode ser longo mas tem que ser único.
Descrição sumária	Uma descrição curta do papel e do propósito do caso de uso
Fluxo de eventos	Uma descrição textual do que o sistema e o utilizador efectuam no contexto do caso de uso de forma compreensível pelos interessados. O fluxo de eventos é estruturado num fluxo básico e sub-fluxos.
Requisitos especiais	Uma descrição textual que reúne todos os requisitos, incluindo os não-funcionais, no caso de uso, que não foram consideradas nos fluxos de eventos, mas que são importantes para o desenho ou implementação.
Pré-condições	Uma descrição textual que define as restrições no sistema quando o caso de uso pode iniciar-se.
Pós-condições	Uma descrição textual que define as restrições no sistema quando o caso de uso termina.
Pontos de extensão	Uma lista de localizações no fluxo de eventos do caso de uso em que podem ser inseridos comportamentos adicionais ou para onde estes podem retomar.
Relações	As relações com outros casos de uso.

A descrição pode ser complementada com diagramas que ilustram aspectos do caso de uso, tais como diagramas de sequência para representar fluxos de eventos ou diagramas de estados para representar o ciclo de vida de entidades.

Os casos de uso podem ser descritos de várias formas variando no seu nível de detalhe e extensão. Na perspectiva de Bittner e Spence [22], as várias formas de escrever casos de uso representam diferentes estados em que este pode estar documentado, desde a sua descoberta inicial até uma descrição muito detalhada. Na tabela 14 apresentam-se os estados e as suas características e objectivos.

Tabela 14 – Estados de documentação de casos de uso

Estados	Característica	Objectivo
1. Descoberto	Indicação apenas do nome e dos actores envolvidos. Ausência de descrição.	Identificar a existência do caso de uso
2. Descrito sumariamente	Inclui uma breve descrição do que o actor pretende fazer com o caso de uso.	Identificar o propósito do caso de uso
3. Esboçado	Identificados os passos do fluxo de eventos do cenário principal. Identificados os cenários alternativos e passos em que ocorrem divergências.	Identificar a forma e extensão do caso de uso
4. Delineado	Descreve o cenário principal sob ponto de vista externo com enfoque deliberado na usabilidade.	Resumir a essência do caso de uso
5. Descrito detalhadamente	Acrescentar detalhes à descrição do cenário principal e detalhar também os cenários alternativos.	Permitir que o detalhe seja adicionado incrementalmente
6. Descrito completamente	Inclui também a lista de todos os requisitos no seu contexto, possibilitando a sua utilização em documentos de especificação de requisitos.	Incluir em especificação de requisitos

Com base na proposta de Bittner e Spence e utilizando a sua escala de estados de documentação, concebeu-se, no âmbito da metodologia de desenvolvimento de requisitos proposta, o formato de estruturação de casos de uso apresentado na tabela 15.

Tabela 15 – Formato da descrição de casos de uso

Identificador	**Nome**
Objectivo	Curta descrição do que é pretendido com o caso de uso.
Descrição sumária	Pequeno texto, com três ou quatro frases no máximo, que descrevam de forma genérica o caso de uso.
Pré-condições	Condições que devem verificar-se no início do fluxo básico do caso de uso.
Pós-condições	Condições que devem verificar-se na conclusão do fluxo básico do caso de uso.
Estado	Descoberto / Descrito sumariamente / Esboçado / Delineado / Descrito detalhadamente / Descrito completamente

Os cenários de utilização são descritos através de fluxos de eventos. Estes podem ser de quatro tipos:

- Fluxo básico – sequência de eventos considerada mais comum para atingir o objectivo do caso de uso (existe um só fluxo básico);
- Fluxos alternativos – outras sequências de eventos para realizar o caso de uso;
- Fluxos opcionais – sequências de eventos que estendem o fluxo básico do caso de uso;
- Fluxos de excepção – sequências de eventos que normalmente levam a que o caso de uso não atinja o seu objectivo.

A descrição dos fluxos de execução acompanham a descrição do caso de uso, contudo, para facilitar a legibilidade, são apresentados em tabelas separadas de acordo com o formato da tabela 16. O fluxo básico é também representado num diagrama de sequência da UML.

Tabela 16 – Formato da descrição de fluxos de eventos

Nome do fluxo		
Passos	**Descrição**	**Pontos de extensão**
Nº do passo	Frase que descreve a interacção entre o actor e o sistema	Indicação opcional do nome do passo

Os pontos de extensão são necessários para referenciação do início e conclusão dos fluxos alternativos, opcionais ou de excepção no fluxo básico.

3.6.2 Formalização de requisitos

A formalização de requisitos (actividade 3.2) é a actividade em que os requisitos da parte do sistema analisada são documentados de forma rigorosa e de acordo com um formato específico. Uma preocupação frequente das equipas de especificação, ao usarem a técnica de modelação de software com casos de uso, é a manutenção de requisitos em mais do que um local: uma expressão no formato de casos de uso e outra na formalização dos requisitos [23]. Essa redundância, além de aumentar o trabalho de manutenção pode levar a inconsistências. Parece prevalecer a ideia de que os casos de uso são utilizados para comunicar com os clientes, enquanto a formalização de requisitos serve para descrever todos os requisitos em grande detalhe e de uma forma bastante declarativa, estruturada e organizada para as equipas de desenvolvimento e testes. No contexto deste problema, Bittner e Spence [22] defendem uma metodologia em que os requisitos devem ser enquadrados nos próprios casos de uso e não mantidos à parte. O seu principal pressuposto é o facto

de não ser simples compreender as formas de utilizar um sistema apenas a partir de uma descrição formal de requisitos.

Na sua perspectiva, as vantagens da combinação de requisitos em casos de uso são as seguintes:

- Clarificação do âmbito – É mais fácil compreender um requisito se estiver enquadrado no contexto de uma funcionalidade, com valor para os utilizadores do sistema, do que em sequência numa lista longa de requisitos.
- Redução de ambiguidade – O caso de uso faculta um contexto de comportamento do sistema que auxilia a reduzir a ambiguidade do requisito.
- Captura da dinâmica do sistema – A descrição dos fluxos de eventos num caso de uso é em si própria considerada como um requisito funcional.
- Facilidade de manutenção – O enquadramento de casos de uso em áreas funcionais de uma organização, tornando mais fácil a manutenção dos requisitos.

Por estes motivos, a formalização de requisitos é uma actividade que só é realizada após a modelação de casos de uso. Sempre que possível, segue-se a recomendação de Bittner e Spence, ou seja, enquadram-se os requisitos em casos de uso. Aqueles em que tal não seja possível vão sendo mantidos à parte.

A norma IEEE 830-1998 [11] recomenda várias formas de estruturar um documento de especificação de requisitos, contudo, não indica nenhum formato para descrever os próprios requisitos. Sobre esse tópico, apenas determina um conjunto de recomendações em que todos os requisitos:

- Devem ser perceptíveis externamente por utilizadores, operadores ou outros sistemas externos;
- Devem incluir uma descrição mínima de cada entrada no sistema, cada saída e todas as funções desempenhadas como resposta a uma entrada ou para apoiar uma saída;
- Devem ser identificados de forma única;
- Devem ser organizados de forma a maximizar a sua legibilidade.

Na tabela 17 apresentam-se os elementos mínimos que, segundo Withall [24], devem estar presentes na descrição de um requisito.

Tabela 17 – Elementos mínimos para a descrição de um requisito

Elemento	Propósito
Identificador de requisito	Identificador único que permite referenciar o requisito de forma desambigua, e dessa forma, manter informação de rastreabilidade.
Prioridade	Indicação da importância e urgência do requisito numa escala pré-definida.
Sumário	Resumo do requisito da forma mais breve possível (entre duas a seis palavras) para melhorar a legibilidade e facilitar a pesquisa de um requisito entre muitos.
Definição	Descrição do requisito utilizando linguagem natural e, se necessário, complementando-a com expressões em linguagem formal, esquemas, diagramas e tabelas.

Embora em muitos casos os quatro elementos mínimos indicados na tabela 17 sejam suficientes, Withall identifica ainda mais alguns elementos adicionais que podem enriquecer a descrição do requisito, indicados na tabela 18. Destes recomenda que sejam incluídos os necessários para desfazer ambiguidades que possam permanecer só com os elementos mínimos.

Tabela 18 – Elementos adicionais para a descrição de um requisito

Elemento	Propósito
Reescrita da definição por outras palavras	Para evitar ao máximo as ambiguidades pode ser benéfico apresentar outra descrição do requisito que utilize outras palavras.
Detalhes adicionais	Alguns requisitos necessitam de detalhes adicionais para apresentar toda a informação necessária.
Exemplos	Os exemplos facilitam a compreensão da ideia transmitida no requisito, mesmo se não forem definidos com muita precisão.
Motivação	Explicação do motivo da inclusão do requisito. Mesmo que a definição seja clara, a motivação da existência do requisito pode não o ser e daí auxiliar a compreendê-lo.
Justificações adicionais para a existência do requisito	Antecipação de apreciações sobre a necessidade do requisito, caso seja previsível que tal possa suscitar discussão.
Resolução de contradições	Se o requisito, no seu todo ou em parte, parecer contradizer outro(s) requisito(s), esses aspectos devem ser clarificados.
Relação com outros requisitos	Referenciar outros requisitos que se relacionem com o discutido e que podem ser úteis para a descrição.
Sugestões de implementação	Podem identificar-se de forma muito breve formas de satisfazer o requisito. Esclarecer que essas sugestões não fazem parte da definição formal do requisito.
Justificação da prioridade	Pode ser necessário explicar porque é que o requisito tem a prioridade que lhe foi atribuída.
Detalhes de eliminação	Se o requisito tiver sido eliminado, indicar porquê, quando e a pedido de quem. Se foi substituído por outro, indicar o seu identificador.

Tanto a norma IEEE 830-1998 como Withall identificam a necessidade de indicar a prioridade dos requisitos de forma a permitir distinguir os mais urgentes dos menos prioritários. Withall propõe a escala de prioridade de cinco níveis indicada na tabela 19.

Tabela 19 – Escala de prioridade de requisitos de Withall

Nível	Significado
Essencial de 1ª prioridade	O sistema não cumpre os seus objectivos se o requisito não for implementado.
Essencial de 2ª prioridade	O sistema pode trabalhar sem o requisito estar implementando mas só durante um período de tempo, podendo implicar inconveniências e apoio manual.
Importante	O sistema tem que satisfazer o requisito para ser considerado completo.
Útil	Representa uma característica que melhora a eficiência operacional do sistema e estende-o para além das funcionalidades essenciais e importantes.
Desejável	Representa uma característica que enriquece o sistema e promove a sua diferenciação relativamente a outros produtos com os quais concorre.

A norma do IEEE, ao descrever o atributo de grau de necessidade de um requisito, propõe a classificação na tabela 20.

Tabela 20 – Graus de necessidade de requisitos pela norma IEEE 830-1998

Grau	Significado
Essencial	O produto não pode ser aceite enquanto o requisito não for implementado da forma descrita.
Condicional	O requisito melhora o produto, mas não o torna inaceitável se não for implementado.
Opcional	Representa uma função que pode ser interessante, dando ao fabricante a oportunidade de propor funcionalidades não previstas na especificação inicial.

Tendo por base o formato de Withall [24] para formalização de requisitos e a escala de graus de prioridade da norma IEEE 830-1998, adoptou-se para a metodologia proposta o formato representado da tabela 21.

Tabela 21 – Formato de formalização de requisitos

Identificador	**Sumário**
Prioridade	Essencial / Condicional / Opcional
Descrição	Texto que descreva o requisito utilizando, se necessário, linguagem formal para que seja compreensível de forma unívoca por todo o público-alvo.
Motivação	Principais razões que levaram à inclusão do requisito. O texto pode ser extenso.
Informação adicional	Detalhes adicionais relacionados com o requisito, mas não essenciais para a sua implementação. O texto pode ser extenso e fazer uso de diagramas, figuras ou outros elementos de apoio.
Sugestões de implementação	Conjunto de sugestões que podem auxiliar a forma de implementação do requisito. O texto pode ser extenso.

A indicação de todos os elementos é obrigatória, excepto os de informação adicional e de sugestões de implementação. O identificador do requisito deve adoptar a recomendação de Withall, ou seja, na forma de um prefixo literal associado a uma área funcional do sistema, seguido de uma numeração sequencial, eventualmente reflectindo uma estrutura hierárquica resultante da decomposição funcional. O sumário é utilizado para o título do requisito e tal como o identificador tem que ser único em todo o documento. As sugestões não se destinam a limitar as opções da equipa de desenvolvimento. Note-se que podem haver requisitos que se destinam a impor condições ao desenvolvimento do sistema, mas nesse caso, o texto para esse efeito é colocado directamente na descrição e não nas sugestões.

O resultado da formalização de requisitos é uma lista dos mesmos apresentados no formato da tabela 21.

Resumindo, a descrição de requisitos, no contexto de cada caso de uso, assume assim duas formas:

- Fluxos de execução – Para capturar a dinâmica da interacção entre os actores e o sistema. Descreve-se sempre o fluxo básico e, se necessário, fluxos alternativos, opcionais ou de excepção. Para mais facilmente se

compreender a interacção entre os actores e o sistema, inclui-se sempre um diagrama de sequência da UML para o fluxo básico.

- Lista de requisitos – Para descrever em detalhe todos os aspectos referenciados nos eventos. Esta lista é incluída sempre depois dos fluxos de execução.

Da conclusão da actividade de formalização de requisitos resulta um artefacto intermédio, designado de especificação parcial, que reúne todos os casos de uso (com fluxos de eventos e requisitos) para a parte do sistema analisada.

3.6.3 Validação da especificação parcial

A validação de requisitos destina-se a efectuar um conjunto de acções correctivas para garantir que o documento tem um nível de qualidade que permita que se possam iniciar as etapas seguintes do ciclo de desenvolvimento de software com vista a construir o sistema pretendido.

A validação de requisitos corresponde à quarta etapa da metodologia de engenharia de requisitos de Kotonya-Sommerville [2]. É descrita nessa metodologia como um processo demorado e que exige a análise e leitura do documento final de requisitos por várias pessoas. Como em outras metodologias, a maior parte do documento é a que diz respeito à descrição dos requisitos do sistema. Na metodologia proposta pretende-se que o documento seja escrito de forma incremental à medida que cada parte do sistema vai sendo analisada e especificada. A actividade de validação de cada especificação parcial (actividade 3.3) permite que esse trabalho possa também ser efectuado de forma incremental, dispensado a necessidade de uma etapa final de validação de todo o documento. Tem como objectivo obter um acordo da equipa de especificação sobre a forma como os requisitos são formalizados. Esta validação não carece de uma reunião formal uma vez que os requisitos já foram negociados e aceites na especificação base.

O engenheiro de requisitos fornece a especificação parcial a cada membro da equipa de especificação e aguarda a recepção de comentários, por exemplo, através de correio electrónico. Os comentários que possam surgir prendem-se apenas com pequenos detalhes e não com a essência do requisito. Se necessário, o engenheiro de requisitos procede a alterações no texto dos requisitos assinalados pela equipa.

3.6.4 Consolidação do documento de requisitos

A actividade de consolidação do documento de requisitos (actividade 3.4) conclui a etapa da sua especificação e documentação e o processo de desenvolvimento dos mesmos. Destina-se a produzir o documento de especificação de requisitos, a partir do estudo prévio e das várias especificações parciais que foram desenvolvidas nas actividades anteriores, para ser entregue às equipas de desenvolvimento e testes. O

material produzido em todas as especificações parciais e parte da informação do estudo prévio é integrada neste documento.

Não existe uma forma única de organizar um documento de especificação de requisitos. No contexto desta metodologia, é proposto o modelo apresentado na tabela 22.

Tabela 22 – Estrutura do documento de especificação de requisitos

Capítulo / secção	Descrição	Origem
Histórico de alterações		
1. Introdução		
1.1. Propósito do documento	Indicação do objectivo e visão geral do documento.	Específico do documento.
1.2. Objectivo do sistema	Enumeração dos objectivos do desenvolvimento do sistema.	Análise de objectivos incluída no documento de estudo prévio.
1.3. Glossário	Definições de termos utilizados no documento.	Consolidação dos glossários incluídos em cada documento de especificação base.
1.4. Referências	Lista de documentos utilizados e citados no documento.	Referencia aos artefactos previamente produzidos e suas referências.
1.5. Formato dos requisitos	Descrição da forma como são apresentados os requisitos.	Formato proposto para a metodologia.
2. Contexto		
2.1. Âmbito	Descrição geral do negócio e ambiente que rodeia o sistema.	Análise da estrutura da organização e processos no documento de estudo prévio.
2.2. Modelo genérico de casos de uso	Diagrama geral de casos de uso do sistema.	Específico do documento.
2.3. Principais pressupostos	Condições necessárias para o sistema ser operacionalizado.	Específico do documento.
2.4. Principais exclusões	Aspectos que não serão contemplados no sistema.	Específico do documento.
3. «nome de área funcional» **…** **(n-1).**	Um capítulo para cada área funcional do negócio com a descrição de casos de uso e formalização de requisitos enquadrados nos casos de uso.	Inclusão dos casos de uso e requisitos de cada especificação parcial.
n. Requisitos suplementares	Formalização de todos os requisitos transversais ao sistema.	Específico do documento.

Para a concepção do modelo adoptado na metodologia de desenvolvimento de requisitos proposta foram analisados três modelos:

- Modelo do IEEE [11] – Modelo proposto na norma IEEE 830-1998 como recomendação de uma forma de organizar um documento de especificação de requisitos;

- Modelo do RUP [25] – Modelo baseado no do IEEE, proposto na metodologia *Rational Unified Process* (RUP) [1] mas com alterações substanciais na forma de descrever o sistema e enquadrar os requisitos em casos de uso;
- Modelo de Withall [24] – Modelo proposto por Withall para a escrita de documentos de especificação de requisitos.

A análise detalhada dos três modelos é apresentada no anexo C.

Esta estrutura é composta por conteúdos recorrentes em cada um dos modelos e adaptado para reflectir a metodologia de descrição de requisitos em casos de uso descrita por Bittner e Spence [22] e sugerida no modelo do RUP. Na mesma tabela indica-se também a origem de alguns conteúdos, indicando quais deles são reutilizados a partir de outros artefactos produzidos previamente.

Esta estrutura sugere que o documento de especificação de requisitos, embora surgindo no final da metodologia de desenvolvimento de requisitos proposta, pode, na realidade, ser produzido de forma incremental à medida que vão sendo disponibilizados artefactos em cada actividade. Ao incluir conteúdos já produzidos anteriormente reduz substancialmente o esforço de escrita do documento.

O documento de especificação de requisitos, dependendo do número de partes do sistema e da sua complexidade, pode ser um documento bastante extenso.

3.7 Conclusões

A metodologia apresentada neste capítulo pode ser encarada como um padrão da forma de realizar processos de desenvolvimento de requisitos para sistemas que se enquadram nos seus pressupostos. Embora possa ser aplicada na prática seguindo as etapas e actividades descritas, é inevitável que cada projecto em particular imponha alterações e pequenas variantes.

No capítulo seguinte apresenta-se um exemplo de aplicação da metodologia à especificação de um sistema de informação.

Capítulo 4

Especificação do sistema WebGA

4.1 Introdução

O sistema WebGA será um sistema de informação que se destina a substituir o sistema GAUP actualmente utilizado no apoio às actividades administrativas do processo pedagógico da Universidade do Porto. No contexto do seu desenvolvimento pretende-se obter o documento de especificação de requisitos que o descreva completamente antes de ser desenhado, implementado e testado. Para esse efeito, é necessário realizar um processo de desenvolvimento de requisitos. O sistema WebGA reúne os pressupostos necessários para aplicação da metodologia de desenvolvimento de requisitos descrita no capítulo anterior. De facto, o sistema WebGA será um sistema de informação de média dimensão, constituído por vários módulos, que suportará processos administrativos e fortemente condicionados por legislação. No seu desenvolvimento foram atribuídas funções a um conjunto de pessoas que implicam a alocação de algumas horas de trabalho semanal para o seu desempenho. Por último, uma dessas pessoas assume a responsabilidade de engenheiro de requisitos com dedicação exclusiva a essas funções.

Neste capítulo descreve-se a aplicação da metodologia na especificação do sistema WebGA. Nas secções, descreve-se o trabalho efectuado no contexto de cada actividade, pela ordem com que foram executadas. Indicam-se os respectivos intervenientes e exemplos dos artefactos produzidos. O objectivo é validar que a metodologia pode ser adoptada para desenvolvimento de requisitos para sistemas com as características do WebGA.

4.2 Identificação de fontes de requisitos e planeamento

4.2.1 Fontes de requisitos identificadas

O sistema WebGA destina-se a apoiar a vertente administrativa dos sub-processos do processo pedagógico de uma instituição de ensino superior. Por esse motivo, para a identificação de fontes de requisitos atendeu-se ao conhecimento que é necessário obter sobre esse processo e tipo de instituições. Sem perder de vista as fontes indicadas pela metodologia proposta, a identificação de fontes de requisitos envolveu a realização das seguintes tarefas:

a) Identificação e caracterização dos interessados;
b) Recolha de legislação sobre o processo pedagógico;
c) Recolha de documentação emitida pela instituição e outros organismos;
d) Identificação dos sistemas existentes de apoio ao processo pedagógico;
e) Recolha dos objectivos gerais do sistema WebGA.

a) Identificação e caracterização dos interessados

Na tabela 23 indicam-se as entidades ou serviços de onde vêm os interessados, as funções de cada interessado e o tipo de envolvimento no sistema WebGA.

Tabela 23 – Interessados no sistema WebGA

Entidade/Serviço	Pessoas	Tipo de envolvimento
Instituições de ensino superior (ex: Universidade do Porto)	Representante	Cliente
Serviços académicos das unidades orgânicas das instituições de ensino superior	Chefe de divisão Técnicos administrativos	Utilização
Projecto de sistemas de informação (PSI) da FEUP	Coordenador Programadores	Desenvolvimento

As instituições de ensino superior representam o cliente do sistema WebGA. Ao adquirirem uma licença de utilização pretendem que o produto satisfaça os objectivos e contribua de forma positiva para melhorar a vertente administrativa do processo pedagógico. É nos serviços académicos das unidades orgânicas das instituições de ensino superior que se encontram as pessoas que utilizarão directamente o sistema. Pretendem que o sistema seja de utilização simples e os auxilie na realização das suas actividades diárias. O PSI é a entidade que coordenará o desenvolvimento do sistema. Este desenvolvimento assegura até ao momento a etapa de especificação de requisitos. O desenvolvimento e testes poderão ser realizados em conjunto com outros serviços da Universidade do Porto.

b) Recolha de legislação sobre o processo pedagógico

Foi pesquisada legislação que regulamenta partes do processo pedagógico de instituições de ensino superior. Parte dela foi fornecida pelos serviços académicos da FEUP enquanto outra foi recolhida por pesquisa na *web*.

Ao recolher-se legislação organizou-se uma tabela com as seguintes colunas:

- Diploma – identificação do tipo de diploma (lei, decreto-lei, deliberação, etc.) e número;
- Título – nome do diploma, na ausência deste, atribuiu-se um título que resuma o seu objectivo;
- Data, nº e série – informação que permite a identificação da publicação do diploma em Diário da República;
- Âmbito – categorização do diploma para organização por tópicos.

A tabela 24 apresenta exemplos da legislação recolhida. À medida que o processo progrediu, verificou-se a necessidade de pesquisar mais legislação específica de cada sub-processo.

Tabela 24 – Exemplos de legislação recolhida

Diploma	Título	Data	Nº	Série	Âmbito
Decreto-Lei nº 42/2005	Princípios reguladores de instrumentos para a criação do espaço europeu de ensino superior	22-02-2005	37	I Série-A	Processo de Bolonha
Lei nº 49/2005	Segunda alteração à Lei de Bases do Sistema Educativo	30-08-2005	166	I Série-A	Missão e objectivos
Deliberação nº 896/2005	Regulamento de aplicação do sistema de créditos curriculares aos cursos conferentes de grau da UP	30-06-2005	124	II Série	Unidades de crédito
Deliberação nº 897/2005	Normas para enquadramento de cursos conferentes de grau na UP	30-06-2005	124	II Série	Criação de cursos
Despacho nº 10543/2005	Caracterização de Cursos	11-05-2005	91	II Série	Gestão de cursos
Decreto-Lei nº 74/2006	Graus Académicos e diplomas do ensino superior	24-03-2006	60	I Série-A	Graus e diplomas
Despacho nº 7287-A/2006	Normas de organização dos processos referentes às alterações de ciclos de estudos	31-03-2006	65	II Série	Cursos e planos de estudo
Lei nº 62/2007	Regime Jurídico das Instituições de Ensino Superior	10-09-2007	174	I Série	Missão e objectivos
Portaria nº 401/2007	Mudança de Curso, Transferências e Reingressos	05-04-2007	68	I Série	Regimes de ingresso

c) Recolha de documentação emitida pela instituição e outros organismos

A documentação de apoio inclui toda a documentação que se considera relevante como fonte de requisitos. Foi efectuada a recolha dos seguintes tipos de documentos de apoio:

- Regulamentos – Interpretam a legislação, focalizando-se nos aspectos que são importantes para a organização;
- Ofícios – Transmitem directivas internas que sobre decisões tomadas por um organismo;
- Normas internas – Norteiam a forma de operacionalizar processos.

Ao recolher-se a documentação de apoio organizou-se uma tabela com as seguintes colunas:

- Tipo – identificação do tipo de documento;
- Título – nome do documento, na ausência deste, atribuiu-se um título que resuma o seu objectivo;
- Data – data de publicação ou produção do documento;
- Nº de páginas – para estimativa do volume do documento;
- Autor – identificação do autor ou organismo que produziu o documento;
- Âmbito – categorização do documento para organização por tópicos.

A maioria dos documentos recolhidos são emitidos pela Universidade do Porto por ser a entidade de mais fácil acesso aos mesmos e simultaneamente representar adequadamente a realidade de uma instituição de ensino superior.

Na tabela 25 apresentam-se alguns exemplos dos documentos recolhidos. À semelhança da legislação, verificou-se a necessidade de pesquisar mais documentos.

Tabela 25 – Exemplos de documentação de apoio

Título	Tipo	Data	Págs.	Autor	Âmbito
Classificação das Áreas Científicas (versão portuguesa do glossário CORDIS)	Regulamento UP	Set-06	38	Reitoria da UP	Gestão de cursos
Número de créditos que cada estudante pode realizar anualmente	Clarificação UP	14-02-2007	1	Senado da UP	Inscrições em disciplinas
Regulamento dos regimes de mudança de curso, transferências e reeingresso da UP	Regulamento UP	05-05-2007	11	Senado da UP	Candidaturas
Proposta de criação de novos cursos na UP	Deliberação UP	28-11-2005	1	Senado da UP	Gestão de cursos
Regulamento geral dos cursos de primeiro ciclo da UP	Regulamento UP	13-09-2006	9	Senado da UP	Gestão de cursos
Regulamento geral dos cursos de segundo ciclo da UP	Regulamento UP	28-09-2006	9	Senado da UP	Gestão de cursos
Regulamento geral dos cursos de terceiro ciclo da UP	Regulamento UP	27-09-2006	9	Senado da UP	Gestão de cursos

Tabela 25 – Exemplos de documentação de apoio (cont.)

Título	Tipo	Data	Págs.	Autor	Âmbito
Normas orientadoras para a criação de programas de dupla/múltipla titulação entre a UP e universidades estrangeiras	Regulamento UP	14-03-2007	3	Senado da UP	
Normas de enquadramento dos cursos conferentes de grau nas unidades orgânicas da UP	Regulamento UP	04-05-2005	4	Senado da UP	Gestão de cursos
Aplicação do sistema de créditos curriculares	Ofício UP	29-11-2007	1	Reitoria da UP	
Aprovação e publicitação de regulamentos de ciclos de estudos	Ofício UP	16-01-2008	1	Reitoria da UP	Gestão de cursos

d) Identificação dos sistemas existentes de apoio ao processo pedagógico

Na identificação dos sistemas existentes utilizou-se mais uma vez como referência o caso da Universidade do Porto. Nesta instituição, os sistemas existentes que apoiam o processo pedagógico são indicados na tabela 26.

Tabela 26 – Sistemas existentes na Universidade do Porto

Sistema	Descrição
GAUP	Sistema que actualmente suporta as partes essenciais da vertente administrativa do processo pedagógico. O sistema WebGA destina-se a substituir este sistema levando a que no âmbito deste processo de desenvolvimento de requisitos, seja encarado como "sistema legado". Utilizado por técnicos administrativos nas unidades orgânicas da Universidade do Porto e em outras instituições de ensino superior.
SIGARRA	Sistema *web* que apoia, entre outras actividades, algumas partes da vertente administrativa do processo pedagógico e interage com o GAUP. Utilizado por estudantes, docentes e pessoal não-docente das unidades orgânicas da Universidade do Porto e de outras instituições de ensino superior.
InfoCiências	Sistema *web* que apoia, entre outras actividades, algumas partes da vertente administrativa do processo pedagógico. Utilizado por estudantes e docentes da Faculdade de Ciências da Universidade do Porto.

e) Identificação dos objectivos gerais do sistema WebGA

Os objectivos do sistema WebGA foram elencados de forma genérica por um representante da Universidade do Porto. Embora tenham de ser refinados e desenvolvidos a partir da interacção com os interessados, os objectivos são suficientemente específicos para justificar o desenvolvimento de um novo sistema.

Representando o sistema WebGA uma evolução simultaneamente técnica e funcional do sistema GAUP, os objectivos delineados podem classificar-se numa dessas categorias, conforme indicado na tabela 27.

Note-se que nesta actividade apenas se indicam os objectivos. Será necessário compreender e aferir de que forma constituem uma evolução do sistema GAUP.

Tabela 27 – Objectivos gerais do sistema WebGA

Objectivo	Categoria
1. Gestão centralizada do processo pedagógico de várias unidades orgânicas	Funcional
2. Adopção do modelo de três camadas no desenvolvimento do sistema	Tecnológico
3. Segurança com dupla protecção	Tecnológico
4. Melhoria de informação de auditoria	Tecnológico
5. Mudança para o paradigma da Web	Tecnológico
6. Suporte da formação contínua	Funcional
7. Integração com outros sistemas	Tecnológico
8. Visão transversal da informação	Funcional
9. Melhoria de funcionalidades	Funcional
10. Produção de documentação	Tecnológico

4.2.2 Equipa de especificação

O representante da Universidade do Porto (que assume a função de moderador) constituiu a equipa de especificação, de acordo com as funções indicadas na tabela 28. Para cada elemento indica-se a entidade de origem, a função que lhe foi atribuída e outras funções que tenha desempenhado relacionadas com outros projectos relacionados com o WebGA que justifiquem a sua inclusão na equipa.

Tabela 28 – Composição da equipa de especificação do sistema WebGA

Membro da equipa	Função atribuída	Entidade de origem	Funções desempenhadas em outros projectos
GD	Moderador	Faculdade de Engenharia da Universidade do Porto	Coordenação do desenvolvimento da 2ª versão do sistema GAUP. Apoio à coordenação do sistema SIGARRA.
MM	Consultor	Projecto de Sistemas de Informação	Coordenação do desenvolvimento da 2ª versão do sistema GAUP. Coordenação do sistema SIGARRA.
MN	Consultor	Projecto de Sistemas de Informação	Colaboração no desenvolvimento do sistema SIGARRA e da 2ª versão do sistema GAUP.
PS	Engenheiro de requisitos	Projecto de Sistemas de Informação	Colaboração no desenvolvimento do sistema SIGARRA e da 2ª versão do sistema GAUP.
EN	Consultor	Reitoria da Universidade do Porto	Colaboração no desenvolvimento da 1ª e 2ª versão do sistema GAUP.
HP	Consultor	Reitoria da Universidade do Porto	Colaboração na unidade de gestão de informação da Universidade do Porto
CB	Consultor	Faculdade de Ciências da Universidade do Porto	Colaboração no desenvolvimento do sistema InfoCiências.

4.2.3 Documento de estudo prévio

Das técnicas apontadas na metodologia proposta foram utilizadas as seguintes:

a) Análise de documentação;
b) Análise de sistemas existentes;
c) Análise de objectivos.

Não foram realizadas entrevistas porque os membros com funções de consultoria possuem o conhecimento necessário sobre o processo pedagógico. A extracção desse conhecimento é remetida para as reuniões de especificação previstas na metodologia.

a) Análise de documentação

Na análise de documentação efectuou-se uma triagem da legislação e outros documentos recolhidos de forma a isolar a que é considerada efectivamente relevante para a especificação do sistema WebGA. Tal implicou a leitura dos documentos o que permitiu extrair parte do conhecimento base sobre:

- Domínio de aplicação do sistema – o processo pedagógico;
- Organização onde será operacionalizado o sistema – a estrutura das instituições de ensino superior.

Estes dois tópicos de análise são as bases que permitem compreender o ambiente em que o sistema será introduzido. O processo pedagógico define o negócio da organização. A estrutura das instituições de ensino superior permite compreender quais são os serviços (ou outras estruturas) que intervêm em cada vertente desse processo.

b) Análise de sistemas existentes

Dos três sistemas existentes foi possível extrair informação sobre o GAUP e o SIGARRA realizando tarefas que envolvem:

- Utilização directa – Navegação, pesquisa de registos, análise de resultados e emissão de relatórios nos sistemas GAUP e SIGARRA.
- Recolha de documentação – Consulta do manual de utilização do sistema GAUP e de um manual técnico que detalha as opções de desenho do mesmo. Consulta de publicações sobre as funcionalidades disponibilizadas no sistema SIGARRA, sobre o seu processo de desenvolvimento e documentos de especificação de requisitos de alguns módulos desenvolvidos.

A informação recolhida por utilização dos sistemas e consulta de documentação associada foi analisada de forma a compreender os sistemas existentes. Para cada um é definida a estrutura física e lógica, sendo de particular importância compreender que partes do processo de pedagógico apoiam. Verificou-se que a estrutura modular de cada sistema está alinhada com cada sub-processo.

c) Análise de objectivos

Para contextualizar os objectivos do novo sistema, isolaram-se três conjuntos a partir da análise: os objectivos da organização, neste caso, instituições de ensino superior, os da informatização da vertente administrativa do processo pedagógico e finalmente os objectivos do novo sistema (figura 24).

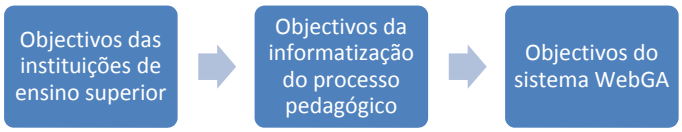

Figura 24 – Decomposição da análise de objectivos

O documento de estudo prévio produzido resulta do tratamento da informação contida nas fontes de requisitos utilizando as técnicas de análise de forma a obter o conhecimento base para dar início à reunião de planeamento.

No caso do sistema WebGA, este documento reuniu a informação resultante da análise de documentação, de sistemas existentes e de objectivos. O documento foi organizado de acordo com os conteúdos apresentados na tabela 29.

Tabela 29 – Visão geral do documento de estudo prévio do sistema WebGA

Capítulo	Conteúdo
1. Introdução	• Enquadramento e descrição do objectivo do documento. • Visão geral da estruturação do documento.
2. Objectivos	• Objectivos genéricos das instituições de ensino superior. • Objectivos da informatização do processo pedagógico. • Objectivos do sistema WebGA.
3. Estrutura da organização e o processo pedagógico	• Descrição detalhada da estrutura da organização, a partir do exemplo da Universidade do Porto. • Análise das diferentes perspectivas que cada interveniente tem do processo pedagógico. • Análise dos fluxos de informação que ocorrem entre os intervenientes na operacionalização do processo pedagógico. • Análise do ciclo do processo pedagógico e identificação dos seus macro-processos e sub-processos.
4. Apoio de tecnologias de informação	• Apresentação dos diferentes níveis de apoio de tecnologias de informação à vertente administrativa do processo pedagógico. • Definição dos conceitos de sistema de informação universitário e de sistemas *front-office* e *back-office*. • Análise detalhada dos sistemas GAUP e SIGARRA a nível de origem, objectivos, exemplos de interface, arquitectura física e arquitectura lógica em que se identificam os módulos que suportam o processo pedagógico. • Esclarecimento do tipo de dependência entre os dois sistemas. • Breve apresentação do sistema InfoCiências. • Breve análise comparativa de sistemas de informação universitários.

4.2.4 Plano de reuniões

A primeira reunião da equipa de especificação, como previsto na metodologia, o objectivo foi a definição do método de trabalho, teve como objectivo a definição do método de trabalho, o particionamento do sistema, atribuição de funções a cada membro da equipa de especificação, discussão dos grandes objectivos do sistema e definição do plano das reuniões seguintes.

Foi criado um *wiki*[10] no SIGARRA para apoiar a especificação do sistema WebGA. O acesso a esse *wiki* está reservado aos membros da equipa de especificação. Qualquer membro pode registar livremente conteúdos e colocar documentos que produza. O plano de reuniões é mantido no *wiki*.

Atendendo à dimensão do sistema WebGA, o esforço de análise e especificação tem que ser particionado. Cada reunião ou grupo de reuniões é dedicada a uma parte do sistema em particular. A definição das partes do sistema, no caso do processo pedagógico, impõe uma ordem natural que segue a ordem com que cada macro-processo descrito no estudo prévio ocorre num ciclo lectivo. Atendendo à dinâmica do processo pedagógico, o particionamento que foi proposto é o representado na figura 25.

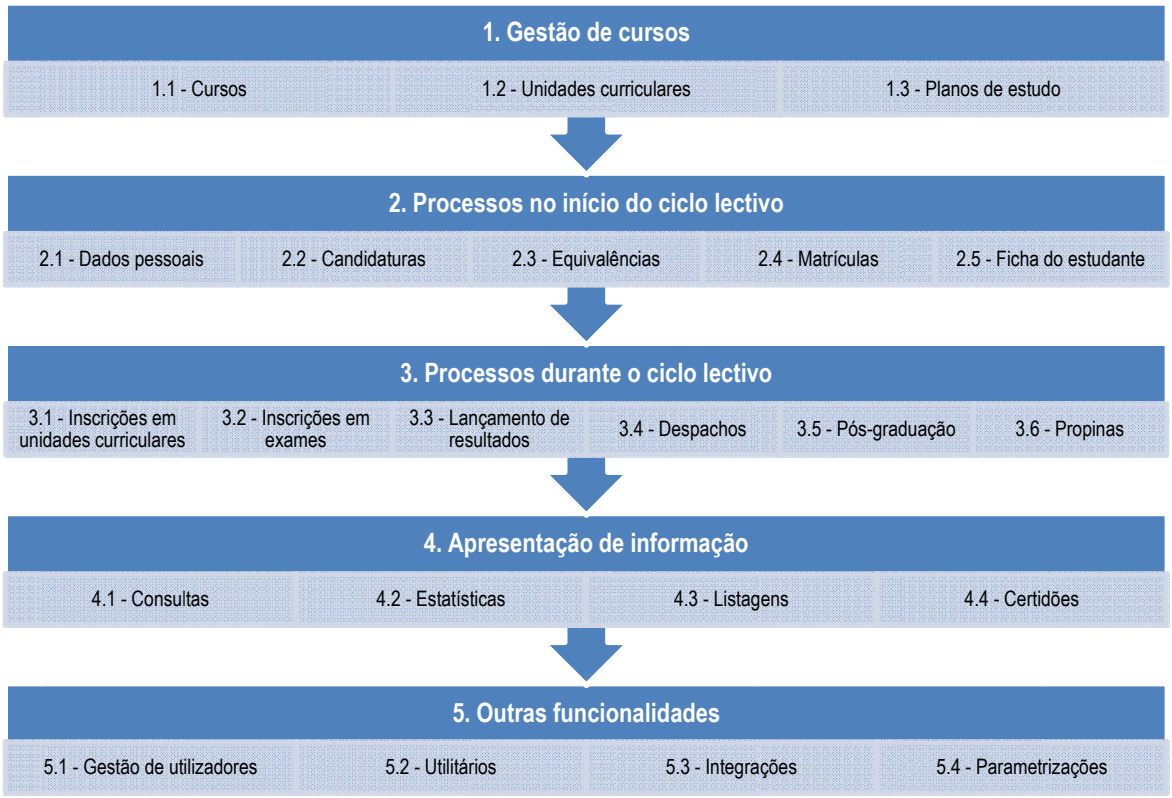

Figura 25 – Particionamento da análise do sistema WebGA

[10] Um *wiki* é um conjunto de páginas *web* interligadas que podem ser consultadas e/ou alteradas de forma simples por qualquer pessoa com permissões para tal.

O particionamento serviu de base para o planeamento de um conjunto de reuniões de especificação e revisão. Até ao momento da escrita deste documento foram agendadas as reuniões indicadas na tabela 30. Para as reuniões já decorridas indicam-se os tópicos debatidos. Pretendeu-se que, sempre que possível, as reuniões decorram com uma frequência semanal, com a duração aproximada de três horas, sem prejuízo de interromper a discussão de tópicos entre reuniões.

Tabela 30 – Plano de reuniões do sistema WebGA

Nº	Data	Parte	Tipo	Resp.	Tópicos debatidos
1	2008-02-21	---	Reunião de planeamento	GD	• Definição do método de trabalho • Definição da equipa • Grandes objectivos do projecto • Planeamento em função das partes da aplicação
2	2008-02-28	Cursos	Reunião de especificação	EN	• Informação mínima de criação de um curso • Designação de um curso, graus e diplomas • Registo de unidade orgânica administrativa • Edições de cursos • Estratégia de replicação e partilha de dados
3	2008-03-04	Cursos	Reunião de especificação	EN	• Estudantes sem curso • Integração dos cursos de educação contínua • Cursos em modo não aprovado
4	2008-03-11	Cursos	Reunião de especificação	EN	• Conceito de tipo de curso • Minors
5	2008-03-20	Cursos	Reunião de especificação	EN	• Informação para o suplemento ao diploma • Estruturas curriculares • Fórmulas
6	2008-04-01	Cursos	Reunião de especificação	EN	• Sistemas de funcionamento
7	2008-04-08	Unidades curriculares	Reunião de especificação	EN	• Definição de terminologia • Análise de elementos caracterizadores
8	2008-04-15	Planos de estudo	Reunião de especificação	MN	• Definição e características genéricas de planos de estudos • Estruturas dos planos de estudo
9	2008-04-22	Planos de estudo	Reunião de especificação	MN	• Estruturas dos planos de estudo (cont.) • Componentes de planos de estudos
10	2008-04-29	Planos de estudos	Reunião de especificação	MN	• Regras de planos de estudos • Atribuição de planos de estudos a estudantes
11	2008-05-13	Dados pessoais	Reunião de especificação	PS	• Análise de elementos relativos a pessoas actualmente registados no sistema de gestão de alunos
12	2008-05-20	Dados pessoais	Reunião de especificação	PS	• Estruturação de moradas • Informação demográfica • Histórico de endereços
13	2008-05-27	Candidaturas	Reunião de especificação	MN	• Regimes de ingresso • Elementos caracterizadores de candidaturas
14	2008-06-03	Cursos	Reunião de validação	PS	• Análise e validação da especificação base de cursos
15	2008-07-01	Cursos	Reunião de validação	PS	• Análise e validação da especificação base de cursos (cont.)

Os tópicos de planeamento, actas e documentos são todos registados em páginas específicas do *wiki*, com ligação a partir da página do plano de reuniões.

4.3 Análise de requisitos por áreas funcionais

4.3.1 Reuniões de especificação

Cada membro da equipa de especificação tem assegurado, com a antecedência necessária, a preparação das reuniões de especificação que lhe foram atribuídas. A preparação do conjunto de reuniões para cada parte do sistema a analisar tem representado, em média, o esforço correspondente a um dia de trabalho. Da preparação resulta uma lista de tópicos que se pretende que sejam discutidos nas reuniões.

A lista de tópicos adopta nenhum formato em particular. Cada membro tem a autonomia para a apresentar da forma que entender mais conveniente. Para a parte referente a cursos foi criado um documento enquanto para as partes restantes já analisadas, a lista foi colocada directamente no *wiki*. A colocação directa no *wiki* tem a vantagem de permitir incluir ligações para outros sítios *web* para referência ao tópico que lhe está associado. Outra vantagem da utilização do *wiki* é que a mesma página pode facilmente servir de base às actas das reuniões, sendo complementada com as decisões tomadas em cada tópico. Na figura 26 apresenta-se um exemplo de utilização do *wiki* na preparação de uma reunião de especificação, concretamente referente à parte de dados pessoais.

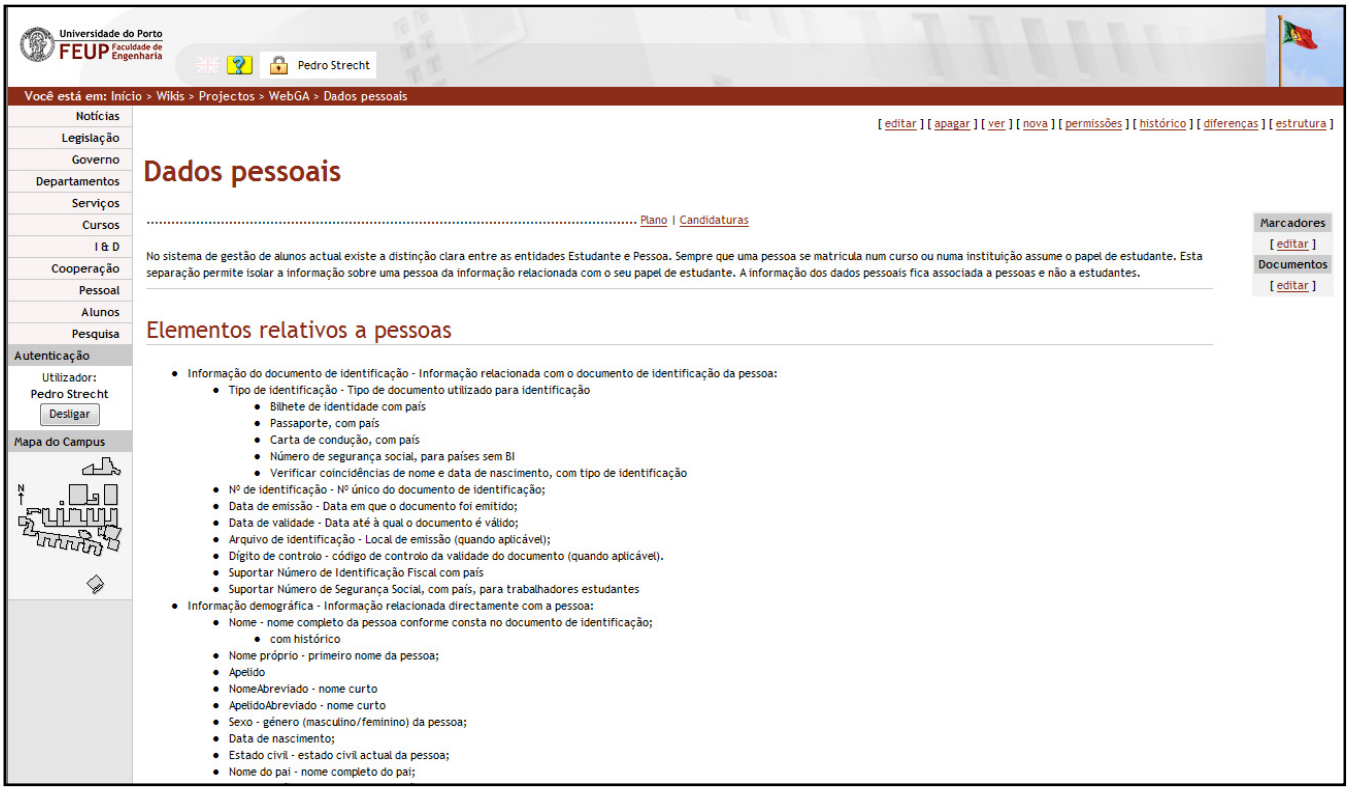

Figura 26 – Utilização de *wiki* para preparação de reuniões de especificação

Como indicado na tabela 30, até ao momento da escrita deste documento já decorreram reuniões de especificação das partes referentes a cursos, unidades curriculares, planos de estudo, dados pessoais e candidaturas. As reuniões têm decorrido da forma descrita pela metodologia proposta com a duração aproximada de três horas. Tem sido possível confirmar que não é produtivo tentar debater assuntos para além dessa duração.

As reuniões têm cumprido eficientemente os seus objectivos, fomentando a discussão aberta mas orientada dos tópicos propostos. A tarefa de elaboração das actas no *wiki* tem sido repartida pelo moderador e pelo engenheiro de requisitos.

Na figura 27 apresenta-se um exemplo do texto de uma acta, concretamente da primeira reunião de especificação da parte referente a cursos.

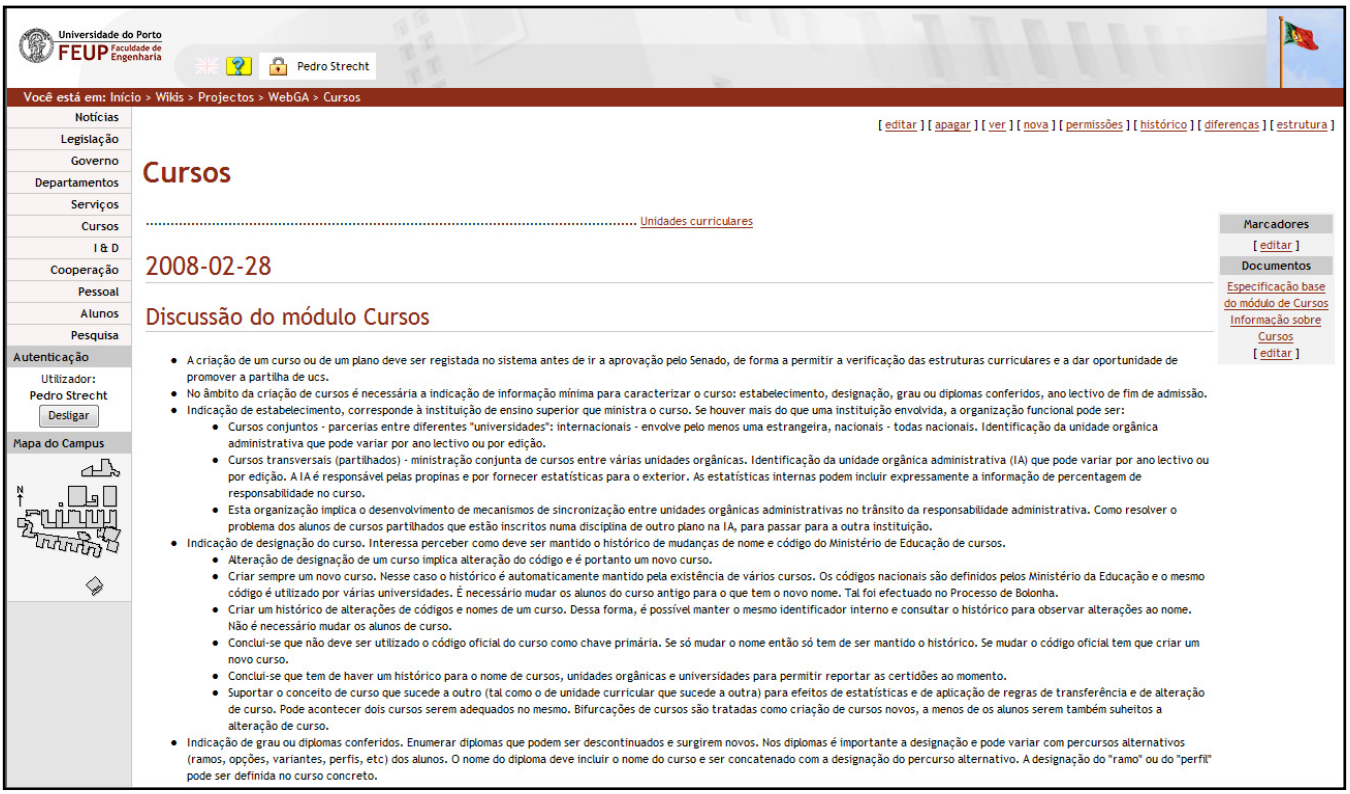

Figura 27 – Utilização de *wiki* para a acta de reunião de especificação

Todas as conclusões definitivas sobre cada tópico discutido têm sido registadas na acta de cada reunião. Se restarem assuntos por discutir por falta de informação são remetidos para uma página do *wiki*, denominada de "assuntos em dossier" criada especificamente para esse efeito. Dentro do possível, essa página deve ser o mais curta possível e idealmente estar vazia no fim do projecto. Sempre que se iniciar uma reunião de especificação de uma nova parte, essa página é inspeccionada uma vez que contém os tópicos que aguardam uma decisão definitiva.

4.3.2 Modelos de negócio produzidos

A metodologia de desenvolvimento de requisitos prevê a elaboração de modelos de negócio a partir das conclusões dos tópicos discutidos nas reuniões de especificação ou da análise das fontes de requisitos para cada parte do sistema. Foram desenvolvidos dois tipos de modelos, utilizando a metodologia de modelação de negócio de Eriksson-Penker [13]:

a) Modelos conceptuais – para organizar, descrever e representar conceitos;
b) Modelos de processos – para representar e descrever processos.

Os diagramas em cada tipo de modelo foram elaborados recorrendo ao IDE de produção de modelo em UML *Enterprise Architect* da Sparx Systems [14] que suporta a notação de Eriksson-Penker.

a) Modelos conceptuais

Os modelos conceptuais foram desenvolvidos de forma a incluírem sempre duas vertentes:

- Mini-glossário, para organizar as definições dos conceitos importantes sobre a parte analisada;
- Diagramas conceptuais, para representar as relações entre os conceitos, utilizando a notação de diagramas de classes, em que:
  - os conceitos são representados como classes, contudo, sem atributos ou métodos;
  - as associações podem ser directas, de generalização ou de composição, podendo ser indicada a multiplicidade;
  - os conceitos genéricos que dependem de uma concretização numa subclasse podem ser representados como classes abstractas.

  Embora os diagramas conceptuais utilizem a mesma notação dos diagramas de classes da UML, importa esclarecer que não se destinam a ser reutilizados da etapa do ciclo de desenvolvimento de software para fins de desenho lógico do sistema.

O modelo conceptual tem sido utilizado como um instrumento de análise que fornece uma perspectiva para identificação célere dos principais conceitos e a forma como se relacionam. Na figura 28 representa-se um exemplo de um diagrama conceptual, concretamente, da parte do sistema referente a planos de estudo.

Até ao momento da escrita deste documento foram desenvolvidos os modelos conceptuais das partes do sistema referentes a cursos, unidades curriculares, planos de estudos e dados pessoais. Para estas partes já foram realizadas reuniões de especificação.

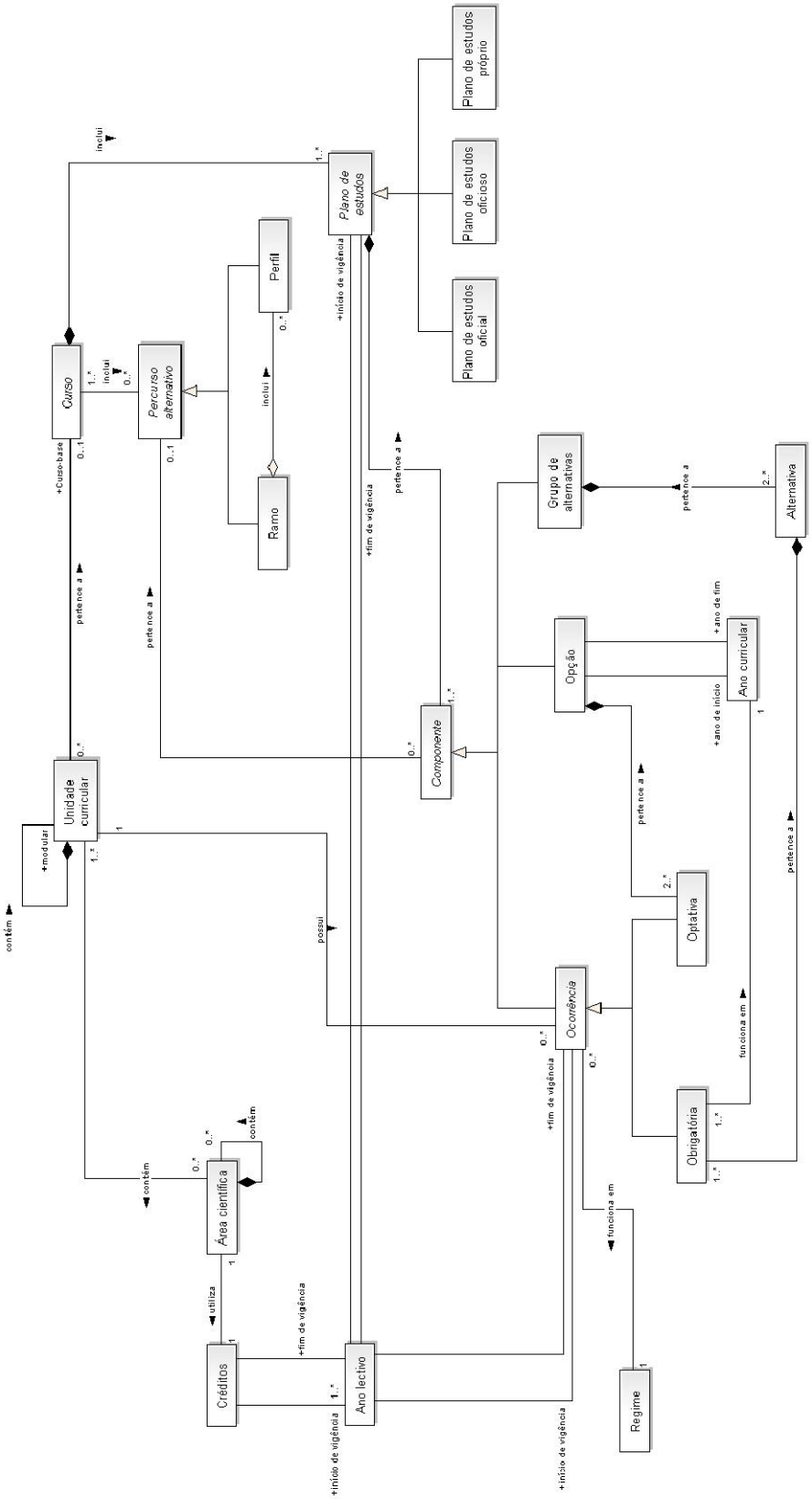

Figura 28 – Diagrama conceptual referente a planos de estudo

a) Modelos de processos

Os modelos de processos apresentam-se sob a forma de diagramas de processos e descrições textuais para representar de forma consistente os principais processos da parte do sistema analisada. A notação dos diagramas de processo da metodologia de Eriksson-Penker resulta da aplicação de estereótipos a diagramas de actividade da UML. A representação inicia-se por uma perspectiva genérica do processo, apresentando-se de seguida uma perspectiva mais detalhada. Para cada processo representa-se:

- o seu objectivo – qual a motivação para a realização do processo;
- os recursos que o controlam – quais os intervenientes que controlam e coordenam o fluxo de execução do processo.
- os recursos que nele participam – outros recursos que são necessários para apoiarem a realização do processo;
- as suas entradas – recursos que serão alvo de transformação durante o processo;
- as suas saídas – recursos que são criados ou alteradas pelo processo.

Os processos são todos representados da mesma forma, assegurando, desse modo, uma análise consistente e completa. O modelo de processos constitui um instrumento de análise que fornece uma perspectiva do processo independentemente da forma como é apoiado por um sistema de informação. Na figura 29 representa-se o diagrama de processos da parte do sistema referente a inscrições em exames.

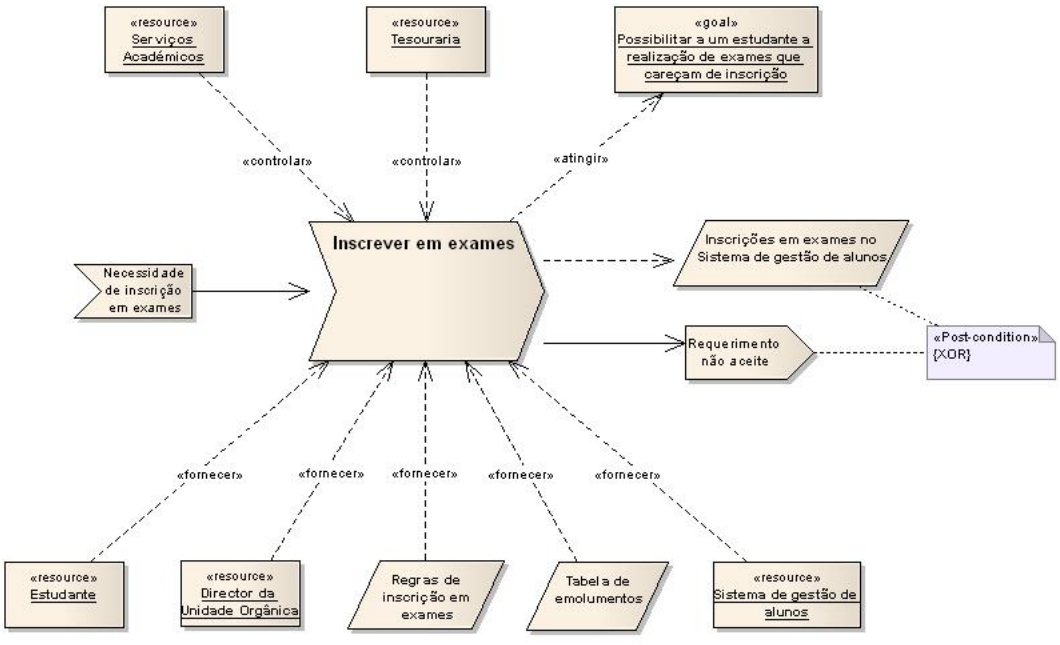

Figura 29 – Diagrama de processo referente a inscrições em exames

O diagrama da figura 29 fornece uma perspectiva de elevado nível de abstracção representando os recursos envolvidos no processo sem detalhar a forma como participam. Esses aspectos são esclarecidos em diagramas mais detalhados em que se representam os sub-processos e actividades em que estão directamente envolvidos.

Cada diagrama de processo, para tornar o modelo completo, tem que ser sempre acompanhado por uma descrição textual que descreva todos os recursos e siga os diferentes fluxos de execução possíveis. A descrição pode ser completada incluindo também diagramas de estados da UML para representar transições de estados de recursos durante o processo.

Até ao momento da escrita deste documento foram elaborados os diagramas de processos das partes do sistema referentes a cursos, equivalências e inscrições em exames. Embora as duas últimas não tenham sido ainda alvo de discussão em reuniões de especificação, foi possível representar os processos por consulta de fontes de requisitos.

4.3.3 Documento de especificação base e reunião de revisão

Já tendo decorrido as reuniões de especificação e elaborados os modelos de negócio do processo referentes à parte de cursos, procedeu-se à preparação da versão preliminar da especificação base referente a essa parte. Esta versão preliminar foi elaborada para ser validada pela equipa de especificação, de forma a comprovar que:

- as conclusões foram correctamente compreendidas;
- os conceitos têm uma definição aceite por todos de forma unânime;
- os processos estão correctamente definidos.

A parte de cursos foi a única para a qual já decorreu uma reunião de revisão. Nessa reunião o documento foi apresentado e exaustivamente analisado pela equipa. Alguns exemplos de alterações indicadas relacionam-se com:

- conclusões que necessitam de uma explicação mais detalhada;
- requisitos que devem ser removidos por interferirem com questões de desenho do sistema;
- raciocínios que devem ser reformulados.

Todas as alterações foram registadas e incorporadas na versão preliminar, dando origem à especificação base definitiva.

Na tabela 31 apresenta-se uma visão geral do documento de especificação base já produzido para a parte do sistema referente a cursos.

Tabela 31 – Visão geral do documento de especificação base referente a cursos

Capítulo	Conteúdo
1. Introdução	• Resumo do documento • Indicação das reuniões decorridas sobre cursos
2. Legislação e regulamentos	• Indicação da legislação aplicável à parte de cursos • Indicação de regulamentos aplicáveis à parte de cursos
3. Tópicos de análise	• Elementos mínimos necessários para a criação de um curso • Unidades orgânicas administrativas e não-administrativas • Denominação de cursos • Percursos alternativos • Graus e diplomas conferidos • Estruturas curriculares e áreas científicas predominantes • Período de funcionamento de um curso • Edição de um curso • Integração da educação contínua • Estudantes sem curso • Modos de um curso • Suplemento ao diploma • Sistemas de funcionamento de um curso
4. Organização de conceitos	• Extracto do glossário referente à parte de cursos • Diagrama conceptual
5. Processos de criação de um novo curso	Descrição do processo com diagramas de processos em diferentes níveis de detalhe, descrição textual e decomposição em sub-processos: • Elaborar e aprovar internamente a proposta de um novo curso • Aprovar externamente e publicar em Diário da República • Nomear os órgãos de gestão do curso • Registar em suporte informático a informação do curso
6. Referências	• Referências utilizadas na escrita do documento.

O capítulo 1 introduz o documento com indicação do seu objectivo e das reuniões de especificação realizadas no âmbito da parte do sistema analisada. No capítulo 2 indicou-se as fontes de requisitos referentes a legislação e regulamentos aplicáveis à parte de cursos. No capítulo 3 apresentam-se os tópicos debatidos nas reuniões de especificação. Sempre que pertinente termina-se cada tópico com um conjunto de requisitos de alto nível. No capítulo 4 apresenta-se o resultado do modelo conceptual, sob a forma do mini-glossário e de um diagrama conceptual elaborado na modelação de conceitos. No capítulo 5 descreve-se o processo de criação de um novo curso através de diagramas de processo acompanhados das respectivas descrições, elaborados na modelação de processos. O processo é decomposto e para cada sub-processo apresenta-se a descrição e respectivo diagrama.

4.4 Especificação e documentação de requisitos

4.4.1 Modelos de casos de uso produzidos

No momento da escrita deste documento foram elaborados alguns casos de uso da parte do sistema referente a cursos. Na figura 30 apresenta-se o respectivo diagrama de casos de uso.

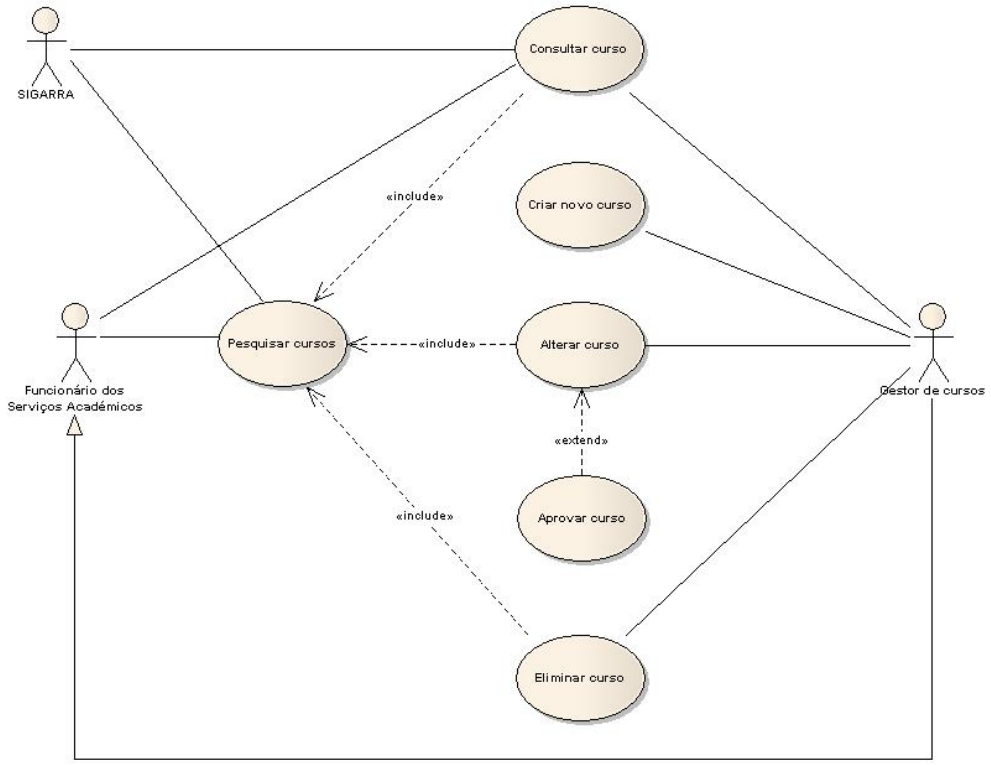

Figura 30 – Diagrama de casos de uso referente a cursos

A descrição de cada caso de uso segue o formato indicado pela metodologia de desenvolvimento de requisitos proposta. Atribuiu-se um identificador a cada caso de uso, constituído por um prefixo indicando a parte do sistema e um número para ordenação. Na tabela 32 exemplifica-se a descrição do caso de uso "pesquisar cursos".

Tabela 32 – Descrição do caso de uso "pesquisar cursos"

CUR.001	**Pesquisar cursos**
Objectivo	Obter cursos registados no sistema.
Descrição sumária	O utilizador indica um conjunto de condições através de critérios apresentados pelo sistema. O sistema obtém os cursos que satisfazem as condições indicadas e apresenta-as ao utilizador. Este pode seleccionar um curso para outras funções.
Pós-condições	Possibilidade de definição ou alteração do *curso seleccionado*.
Estado	Descrito completamente.

Como descrito na metodologia, a seguir à descrição de cada caso de uso indicam-se os fluxos de eventos possíveis. Na tabela 33 exemplifica-se a descrição do fluxo básico do caso de uso "pesquisar cursos", de acordo com o formato definido.

Tabela 33 – Fluxo básico do caso de uso "pesquisar cursos"

Fluxo básico		
Passo	**Descrição**	**Ponto de extensão**
1	O utilizador solicita ao sistema a pesquisa de cursos.	
2	O sistema apresenta ao utilizador um formulário para indicação dos critérios de pesquisa de cursos.	
3	O utilizador indica uma ou mais condições para restringir os cursos pretendidos.	**Condições**
4	O sistema obtém os cursos que satisfazem todas as condições indicadas.	**Obter cursos**
5	O sistema apresenta informações sobre os cursos obtidos.	**Apresentar cursos**
6	O utilizador selecciona um dos cursos apresentados.	**Seleccionar**
7	O sistema assume o curso como *curso seleccionado*.	
	Conclusão do caso de uso atingindo o objectivo.	

A captura dos requisitos é obtida directamente dos eventos dos fluxos de execução do caso de uso. Por análise dos eventos do fluxo básico do exemplo da tabela 33, é imediatamente perceptível a necessidade de descrever requisitos que esclareçam:

- quais são os critérios de pesquisa de cursos;
- como é combinada a informação dos critérios;
- como pode o utilizador indicar condições sobre esses critérios;
- como deve ser apresentada a informação.

Na tabela 34 exemplifica-se a descrição de um destes requisitos, de acordo com o formato definido na metodologia.

Tabela 34 – Requisito do caso de uso "pesquisar cursos"

REQ.CUR.0120	**Interface de apresentação de resultados da pesquisa de cursos**
Prioridade	Essencial
Descrição	Os resultados da pesquisa de cursos são apresentados mediante uma tabela com as seguintes colunas: • Código do curso; • Designação do curso; • Sigla do curso; • Unidade orgânica que ministra o curso; • Período (Ano lectivo de início e de fim do curso); • Máximo grau conferido; • Modo de curso; • Tipo de curso. Por omissão, os cursos são ordenados por designação. Deve ser possível ordenar de forma ascendente e descendente por qualquer coluna. A tabela não deve apresentar mais de 20 cursos de cada vez. Deve ser possível solicitar os 20 cursos seguintes e anteriores aos apresentados. O utilizador deve dispor de um mecanismo para seleccionar um curso. A informação sobre essa selecção deve manter-se disponível e constituir o *curso seleccionado*. Ao seleccionar um curso, o utilizador tem acesso à consulta de informações sobre o mesmo (REQ.CUR.200)

4.4.2 Especificações parciais produzidas

No momento da escrita deste documento estão descritos alguns requisitos da parte do sistema referente a cursos, pelo que a respectiva especificação parcial ainda não está completa. Como indicado pela metodologia, pretende-se isolar para cada um desses artefactos intermédios os casos de uso (incluindo fluxos de execução e requisitos) de cada parte do sistema analisada. Como ainda não está completa, a especificação parcial referente a cursos ainda não foi validada pela equipa de especificação.

4.4.3 Documento de especificação de requisitos

Embora ainda não exista nenhuma especificação parcial finalizada, iniciou-se a escrita do documento de especificação de requisitos para validar a metodologia.

O documento segue a estrutura proposta pela metodologia e resulta da consolidação de informação já registada no documento de estudo prévio, da inclusão dos casos de uso já desenvolvidos para a parte de cursos e do desenvolvimento de requisitos suplementares.

A redacção do documento iniciou-se pelo capítulo de "Introdução". Na secção "Propósito do documento" esclarece-se o objectivo de um documento de especificação de requisitos, identifica-se o público-alvo do documento em questão e apresenta-se uma visão geral do resto do documento. Na secção "Objectivo do sistema" enquadra-se o sistema WebGA como sistema que substitui o sistema GAUP. Reconhecendo que o sistema WebGA tem que assegurar todas as funcionalidades já disponíveis no sistema GAUP, considerou-se pertinente apresentar os objectivos delineados para este em 1992. Seguidamente, apresentam-se os principais objectivos do sistema WebGA distinguindo os que induzem evolução do ponto de vista funcional dos que induzem evolução do ponto de vista tecnológico. Note-se que estes objectivos (tanto os do GAUP como os do WebGA) já tinham sido apresentados no documento de estudo prévio. Na secção "Glossário" compilaram-se os mini-glossários de cada especificação base que integra, neste momento, a informação referente a cursos, unidades curriculares, planos de estudos e dados pessoais. Na secção "Referências" incluem-se os recursos consultados e que são referenciados ao longo do documento. Na secção "Formato dos requisitos" apresenta-se o formato com que são descritos os casos de uso, fluxos de execução e requisitos. Clarificam-se também a escala de estados de descrição de casos de uso e os graus de prioridade dos requisitos.

Na redacção do capítulo de "Contexto" foram aproveitados alguns conteúdos do documento de estudo prévio para a secção de "Âmbito". Contudo, outros originais foram também produzidos. Esta secção foi dividida em quatro subsecções, como indicado na tabela 35.

Tabela 35 – Subsecções da secção "Âmbito"

Sub-secção	Conteúdo
1. Perspectivas do processo pedagógico numa instituição de ensino superior	Relação entre as estruturas organizativas com responsabilidades no processo pedagógico (conselho pedagógico, serviços académicos e departamentos) com a respectiva perspectiva do processo pedagógico (estratégica, administrativa e operacional) indicando também alguns exemplos de actividades que realizam. Esta análise é um resumo da apresentada no estudo prévio.
2. Principais fluxos de informação no processo pedagógico	Representação num diagrama de tipo *mind map*[11] dos principais fluxos de informação entre as estruturas organizativas esclarecendo o tipo de actividades que os activam. Esta análise é um resumo da apresentada no estudo prévio.
3. Ciclo do processo pedagógico	Representação do ciclo do processo pedagógico recorrendo a um esquema (utilizado no estudo prévio) e a um diagrama de processo da metodologia de Eriksson-Penker, concluindo que o processo é realimentado.
4. O sistema WebGA no contexto do processo pedagógico	Identificação dos principais sistemas que apoiam actualmente o processo pedagógico (GAUP e SIGARA) e enquadramento do sistema WebGA recorrendo a um diagrama de contexto como sugerido por Withall. Inclusão da análise de mecanismos de comunicação entre o SIGARRA e o GAUP efectuada no estudo prévio.

Na secção seguinte é apresentado o "modelo genérico de casos de uso". Atendendo à dimensão do sistema WebGA seria impraticável apresentar, mesmo que de uma forma resumida, todos os casos de uso previsíveis. Alternativamente, optou-se por apresentar um diagrama de pacotes. Neste diagrama, representado na figura 31, cada pacote agrega uma ou mais partes do sistema que se destinam a suportar sub-processos do processo pedagógico e/ou a reunir um conjunto de funcionalidades. Em cada pacote incluem-se os actores e alguns exemplos de casos de uso previstos. São representadas igualmente as relações de dependência entre os pacotes (por exemplo, o pacote de planos de estudos depende do pacote de cursos). Os actores são somente incluídos nos pacotes em que surgem pela primeira vez. Assume-se que os pacotes que não têm actores, na realidade, incluem os actores dos pacotes de que dependem. Os pacotes para os quais não são indicadas dependências são pacotes com casos de uso transversais a todo o sistema. Na mesma secção descrevem-se ainda cada um dos actores e pacotes.

Ao contrário do indicado no formato proposto para o documento de especificação de requisitos, não foi incluída a secção "Principais pressupostos". Tal deve-se ao facto de o sistema WebGA não ter dependências externas ou condicionalismos exteriores. No que diz respeito à infra-estrutura de suporte, trata-

[11] Diagramas de *mind map* são utilizados para representar palavras, ideias, tarefas ou outros elementos organizados radialmente em redor de uma palavra-chave ou ideia (não são parte da UML).

se da mesma que suporta actualmente o sistema SIGARRA pelo que não impõe novas necessidades que justifiquem a sua referência.

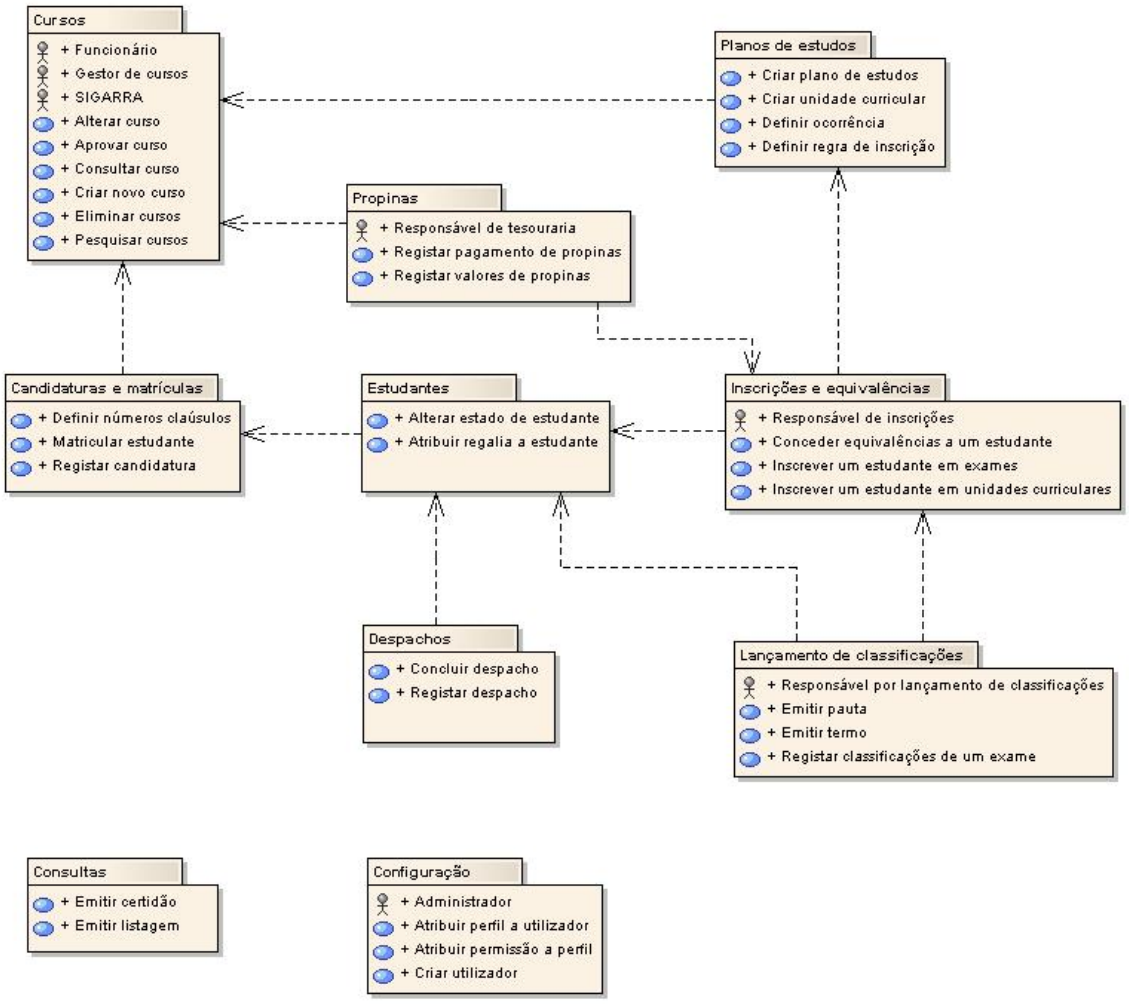

Figura 31 – Diagrama de pacotes do sistema WebGA

O capítulo de "Âmbito" é encerrado com a secção "Principais exclusões" em que são apresentados os sub-processos que não serão contemplados no sistema WebGA. A justificação para as exclusões relaciona-se com o facto de necessitarem de informação de outros sistemas (por exemplo: recursos humanos e instalações) e já serem suportados pelo sistema SIGARRA (que interliga vários sistemas).

Como indicado pela metodologia, os capítulos seguintes dedicam-se à especificação em detalhe dos requisitos de cada parte do sistema. Em vez de haver um mapeamento directo entre partes do sistema (conforme analisadas na metodologia) e capítulos, optou-se por utilizar a decomposição em pacotes apresentada na secção "modelo genérico de casos de uso". No momento da escrita deste documento foram incluídos os casos de uso da especificação parcial da parte

de cursos para o capítulo referente ao pacote de cursos. Prevê-se que os capítulos restantes resultem da mera compilação das especificações parciais que venham a ser produzidas.

O capítulo final do documento consiste numa lista de requisitos suplementares, ou seja, requisitos transversais a todo o sistema e por isso não enquadráveis em nenhum caso de uso em particular. Esta lista de requisitos está a ser mantida no próprio documento de especificação de requisitos. A sua organização baseia-se nos padrões de requisitos propostos por Withall [24] dos quais se indicam alguns exemplos na tabela 36. A lista de requisitos suplementares vai sendo incrementada com novos requisitos à medida que prossegue a análise de partes do sistema e se vai aumentando o conhecimento sobre os requisitos não-funcionais.

Tabela 36 – Alguns padrões de requisitos de Withall

Nome	Situação de utilização
Interface entre sistemas	Para especificar os detalhes básicos para a interface entre o sistema a ser especificado e outro sistema ou componente externo com o qual tenha de interagir.
Interacção entre sistemas	Para especificar um tipo de interacção em particular para interfaces entre vários sistemas.
Tecnologia	Para especificar a tecnologia que deve (ou não) ser utilizada para construir ou executar o sistema, ou com a qual o sistema deve ser capaz de interagir ou assegurar compatibilidade.
Conformidade com normas	Para especificar que o sistema deve estar em conformidade com uma norma ou padrão em particular.
Referência a requisitos	Para especificar que alguns ou todos os requisitos numa especificação de requisitos externa têm que ser satisfeitos tal como se estivessem incluídos no documento.
Documentação	Para especificar tipos de documento de apoio específicos.
Tempo de resposta	Para especificar quanto tempo o sistema pode demorar a dar uma resposta a um pedido.
Taxa de saída	Para especificar a taxa à qual o sistema tem que ser capaz de desempenhar um processamento específico.
Capacidade dinâmica	Para especificar o número de instâncias de um tipo de entidade em particular para a qual o sistema tem que ser capaz de efectuar processamentos em simultâneo.
Capacidade estática	Para especificar a quantidade de um tipo de entidade em particular que o sistema tem que ser capaz de armazenar de forma persistente em sistemas de gestão de bases de dados.
Disponibilidade	Para definir o período em que o sistema tem que estar disponível para os utilizadores ou outros sistemas que dele dependam.
Expansibilidade	Para especificar a forma como o sistema deve acomodar alterações sem implicar reestruturação, tipicamente resultante de aumentos de volume de negócio ou novos processos de negócio.
Extensibilidade	Para obrigar a que um aspecto específico do sistema seja construído de forma a tornar simples a sua extensão mediante acoplamento de software adicional.
Versatilidade	Para especificar que o sistema tem de acomodar a execução de várias sessões em simultâneo, cada uma com interfaces distintas e com isolamento rigoroso de dados entre si.
Suporte de várias línguas	Para especificar que o sistema tem de ser capaz de apresentar interfaces em mais do que uma língua.
Condições de instalação	Para especificar a forma de instalar ou actualizar o sistema.
Registo de utilizadores	Para especificar como se registam novos utilizadores no sistema e detalhes de formas de autenticação.

4.5 Conclusões

A tabela 37 apresenta um resumo das actividades realizadas e artefactos produzidos durante a realização do processo de desenvolvimento de requisitos.

Tabela 37 – Resumo de actividades desenvolvidas

Etapas	Actividade e sub-actividades	Artefactos produzidos	Partes analisadas				
			Cursos	Unidades curriculares	Planos de estudos	Dados pessoais	Candidaturas
1. Identificação de fontes de requisitos e planeamento	1.1 Identificação das fontes de requisitos • Identificação dos interessados; • Identificação de legislação e regulamentos; • Identificação dos sistemas GAUP e SIGARRA; • Identificação dos objectivos do sistema WebGA.						
	1.2 Constituição da equipa de especificação						
	1.3 Elaboração do estudo prévio • Análise de documentação; • Análise de sistemas; • Análise de objectivos.	• Estudo prévio					
	1.4 Realização da reunião de planeamento • Definições das funções da equipa; • Particionamento do sistema; • Planeamento.	• Plano de reuniões					
2. Análise por áreas funcionais	2.1 Preparação de reuniões de especificação	• Listas de tópicos	x	x	x	x	x
	2.2 Realização de reuniões de especificação	• Actas de reuniões	x	x	x	x	x
	2.3 Modelação da área de negócio • Modelação de conceitos; • Modelação de processos.	• Modelos de conceitos	x	x	x	x	
		• Modelos de processos	x				
	2.4 Elaboração da versão preliminar da especificação base	• Versões preliminares de especificação base	x	x	x	x	
	2.5 Realização de reuniões de revisão • Alterações à versão preliminar de especificação base.	• Especificação base definitiva	x				
3. Especificação e documentação de requisitos	3.1 Modelação de casos de uso • Descoberta de casos de uso; • Descrição de fluxos de execução.	• Especificação parcial (incompleta)	x				
	3.2 Formalização de requisitos						
	3.3 Validação da especificação parcial (não efectuado)						
	3.4 Consolidação do documento de requisitos • Compilação de glossário • Descrição do âmbito • Modelação genérica de casos de uso • Descrição de principais exclusões • Compilação da especificação parcial • Formalização de requisitos suplementares	• Especificação de requisitos (incompleto)					

Decorridos seis meses desde o início da especificação foi já possível realizar pelo menos uma vez quase todas as actividades previstas na metodologia. A primeira etapa foi concluída, obtendo-se o estudo prévio, o plano de reuniões e a identificação de grande parte das fontes de requisitos. A segunda etapa foi exercida para cinco partes do sistema, embora com diferentes estados de completude. A parte referente a cursos foi aquela em que foi possível realizar mais actividades. A terceira etapa foi a menos concretizada, havendo a produção de versões incompletas tanto da especificação parcial referente à parte de cursos como do documento de especificação de requisitos.

Capítulo 5

Conclusões e trabalho futuro

5.1 Resultados alcançados

A metodologia de desenvolvimento de requisitos de um sistema de informação conjugou a introdução de aspectos inovadores com a reutilização de aspectos de outras metodologias.

Foi implementada uma decomposição do processo de desenvolvimento de requisitos em etapas e actividades baseada na metodologia de Kotonya-Sommerville [2]. Esta decomposição permite estruturar a metodologia proposta, obtendo uma ordem de realização de actividades.

Foi implementado um particionamento do sistema, que permite a análise pormenorizada de cada parte do sistema em especificação, o que não só reduz a sua complexidade como também permite a execução de diversas actividades paralelamente. Regra geral, esta abordagem introduz uma redução do tempo de análise do sistema.

Foi introduzido um ambiente colaborativo na análise de requisitos, que permite obter uma perspectiva mais representativa da área de negócio através da observação sistemática realizada por uma equipa de especificação.

Foram definidos artefactos intermédios (estudo prévio e especificações base), que permitem separar os requisitos do sistema da informação de apoio.

Foi utilizada a modelação de negócio como instrumento de análise de requisitos, recorrendo à metodologia de Eriksson-Penker [13], que permite enquadrar a contribuição do sistema no negócio em que insere e apontar novas formas de integração.

Foi adoptada a recomendação de Bittner e Spence [22] para enquadramento de requisitos funcionais em casos de uso, aproveitando os fluxos de eventos para descrever a dinâmica do sistema, o que facilita a utilização dos requisitos nas etapas seguintes do ciclo de desenvolvimento de software.

Foi elaborado um novo modelo de documento de especificação de requisitos tendo por base a análise de três modelos: norma IEEE 830-1998 [11], metodologia de desenvolvimento de software RUP [1] e Withall [24].

Foi aplicada a metodologia desenvolvida ao sistema WebGA. Apesar de este sistema assentar nos mesmos pressupostos que a metodologia desenvolvida, a sua aplicação a este sistema de informação específico acabou por revelar que algumas actividades não necessitavam de ser implementadas. Tal permite concluir que, apesar da sua especificidade, esta metodologia necessita ainda de algum nível de adaptabilidade.

Foi utilizada uma ferramenta CASE para a produção de diagramas. A sua utilização revelou-se praticamente indispensável pelo modo como não só agiliza o seu desenho como permite validar o resultado obtido quer a nível sintáctico como a nível semântico. De facto, é difícil manter os requisitos e casos de uso num documento de texto.

Foi utilizada uma ferramenta de carácter inerentemente colaborativo: o *wiki*. Tal ferramenta revelou-se uma excelente plataforma que se alinha com a filosofia da metodologia proposta. Além de servir de repositório dos artefactos produzidos permite monitorizar o estado do processo.

5.2 Sugestões para trabalho futuro

A utilização de um *wiki* apenas permite registar informações como o plano, a ordem de trabalhos ou a acta das reuniões de especificação e validação de requisitos. Porém não permite a produção automática de artefactos resultantes da metodologia.

Existe a necessidade de desenvolver uma funcionalidade da aplicação de gestão de projectos do PSI que constitua uma ferramenta de geração automática de um documento de especificação de requisitos para um sistema ou módulo. A utilização desta ferramenta permitirá indicar conteúdos menos estruturados e automatizar os mais estruturados, como casos de uso, fluxos de execução e requisitos. A ferramenta poderá também apoiar a produção de outros artefactos como o estudo prévio e especificação base.

Foi identificada no PSI a necessidade de desenvolvimento de uma metodologia para realização da etapa de verificação e validação no ciclo de desenvolvimento de software. Um alinhamento com a metodologia de desenvolvimento de requisitos proposta conduz à adopção de uma abordagem *test-driven specification* em que se prevê que os casos de teste sejam concebidos durante a especificação de requisitos.

Referências

1. Software, R., *Rational Unified Process - Best Practices for Software Development Teams*. 1998.
2. Sommerville, G.K.a.I., *Requirements Engineering: Processes and Techniques*. 1998.
3. Donn Le Vie, J. (2007) *Writing Software Requirements Specification*.
4. Patton, R., *Software Testing*. 2001.
5. McConnell, S., *Software Project Survival Guide*. 1998.
6. Wiegers, K.E., *Software Requirements*. 2003.
7. Leonard Tripp, A.A., James W. Moore, Pierre Bourque, Robert Dupuis, *Software Engineering Body of Knowledge*. 2004.
8. *UML® Resource Page*. [acedido em 2008-04-29]; Disponível em: http://www.uml.org/.
9. Benjamin Zeiss, D.V., Ina Schieferdecker, Helmut Neukirchen, Jens Grabowski (2007) *Applying the ISO 9126 Quality Model to Test Specifications*.
10. Japenga, R. (2003) *How to write a software requirements specification*.
11. IEEE, *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications -Description*. 1998.
12. Burnstein, I., *Practical Software Testing*. 2003.
13. Hans-Erik Eriksson, M.P., *Business Modeling with UML: Business Patterns at Work*, ed. I. John Wiley & Sons. 2000.
14. Systems, S., *The Business Process Model*. 2007.
15. Judith Barrios, S.N., *Model Driven Architectures for Enterprise Information Systems*. 2004.
16. Marshall, C., *Enterprise Modeling with UML: Designing Successful Software through Business Analysis*. 1999: Addison-Wesley Professional.
17. Barrios, J.M.a.J., *BMM: A Business Modeling Method for Information Systems Development*. Clei Electronic Jorunal, Vol. 7, nº 2, Paper 3, 2004.
18. N. S. Nagaraj, S.T.S.B., *Business Process Management: An Emerging Trend*. 2001.
19. Cockburn, A., *Use cases, ten years later*. STQE Magazine, 2002.
20. Gorman, J., *Use cases - An Introduction*. 2006.
21. Ruth Malan, D.B., *Functional Requeriments and Use Cases*. 2001.
22. Kurt Bittner, I.S., *Use Case Modeling*. 2002: Addison Wesley.
23. Leslee Probasco, D.L., *Combining Software Requirements Specification with Use-Case Modeling*.
24. Withall, S., *Software Requirements Patterns*. 2007: Microsoft Press.

25. *Rational Unified Process Software Requirements Specification Template.* [acedido em 2008-06-29]; Disponível em: http://www.ts.mah.se/RUP/RationalUnifiedProcess/webtmpl/templates/req/rup_srsuc.htm.
26. Taylor, D.A., *Business Engineering With Object Technology*. 1995: John Wiley & Sons.
27. White, S.A., *Business Process Modeling Notation (BPMN), Version 1.0 - May 3, 2004*. 2004.
28. Arkin, A., *Business Process Modeling Language (BPML) - Specification Draft.* 2002.
29. Tony Andrews, F.C., Hitesh Dholakia, Yaron Goland, Johannes Klein, Frank Leymann, Kevin Liu, Dieter Roller, Doug Smith, Satish Thatte, Ivana Trickovic, Sanjiva Weerawarana, *Business Process Execution Language for Web Services, Version 1.1 - 5 May 2003*. 2003.
30. Group, O.M., *Object Constraint Language - UML 2.0 OCL Specification*. 2003.
31. Cockburn, A., *Structuring use cases with goals*. Journal of Object-Oriented Programming, 1997.

Anexo A

Metodologia de modelação de negócio de Eriksson-Penker

5.3 A.1 Introdução

A metodologia de modelação de negócio de Eriksson-Penker é apresentada no livro "*Business Modeling with UML: Business Patterns at Work*" da autoria de Hans-Erik Eriksson e Magnus Penker.

Na perspectiva da metodologia de Eriksson-Penker, a UML foi inicialmente concebida para descrever aspectos de um sistema de software e, por esse motivo, não representa claramente conceitos de negócio como processos, objectivos, recursos e regras de negócio. Para permitir que a UML também possa ser utilizada na modelação de negócios, foram criadas as extensões de negócio de Eriksson-Penker que oferecem notação apropriada para esse fim. Os mecanismos de extensão da UML utilizados para permitir a acomodação destes novos conceitos são:

- Estereótipos – Criam novos blocos de construção a partir dos já existentes mas específicos de uma área de aplicação em concreto. Podem introduzir ícones novos.
- Valores etiquetados (propriedades) – Criam novas propriedades num elemento UML, permitindo incluir nova informação na especificação desse elemento.
- Restrições – Criam novas regras dos elementos da UML, permitindo aumentar a sua semântica.

Na secção seguinte apresenta-se o enquadramento da metodologia de Eriksson-Penker noutras metodologias de modelação de negócio. Nas restantes secções apresenta-se um resumo da metodologia e dos mecanismos de modelação que propõe.

5.4 A.2 Metodologias de modelação de negócio

Os esforços de aproximação dos conceitos de negócio aos do desenvolvimento de software tiveram as suas origens no paradigma de programação orientada por objectos. A definição de classes e a instanciação de objectos procurava emular a realidade complexa do negócio que os sistemas suportavam. Algumas metodologias como a de Taylor [26], apresentada em 1995, propõem a ideia de um modelo único, defendendo que o modelo do negócio deve ser implementado directamente no software. Na sua concepção de "engenharia convergente", o modelo, embora único, pode ser visto sob duas perspectivas: a do negócio e a do software. Contudo estas encontram-se alinhadas. Na metodologia *Enterprise Knowledge Development* (EKD) [15], desenvolvida por Bubenko, Loucopoulos e Rolland em 1996, propõe-se uma colecção de métodos, técnicas e ferramentas para a análise, planeamento, desenho e propostas de alteração de um negócio. O seu objectivo era a produção de modelos do negócio completos e consistentes, porém, ao contrário da anterior, reconhece não ser uma abordagem que levará necessariamente a modelos de software.

Quando a *Unified Modeling Language* (UML) [8] foi proposta, em 1997, havia uma proliferação de padrões e notações para criar modelos de sistemas de software. Por esse motivo, a UML teve como grande objectivo tornar-se a linguagem padrão que reunisse as características mais bem sucedidas de outras notações. Paralelamente existiam também diversas notações para criar modelos de negócio. Com o amadurecimento dos processos de engenharia de requisitos de sistemas de informação, a modelação de negócio passou a tornar-se uma parte integrante da descoberta e análise de requisitos. Esta integração justifica-se pelo facto de, em muitas organizações, os sistemas de informação já não se limitarem a suportar o negócio. O seu nível de envolvimento implica que passem a fazer parte do próprio negócio e sejam encarados como um recurso estratégico. Nesta perspectiva, é o negócio que orienta os requisitos dos sistemas de informação e pressiona-os para que se aproximem deste o mais possível. Em sequência do reconhecimento da necessidade de incorporar a representação de partes do negócio nos modelos de software, começaram a surgir novas metodologias que utilizavam a notação da UML para esse efeito.

Uma das metodologias baseadas em UML mais utilizadas é o *Rational Unified Process* (RUP) [1] proposto pela IBM. Apesar de ser uma metodologia de engenharia de software, focaliza-se nos processos de negócio e nos objectos de negócio. Defende que o desenvolvimento de um sistema de software deve iniciar-se com a modelação de casos de uso para obter os requisitos funcionais de um sistema. Um caso de uso descreve uma utilização específica do sistema por um ou mais actores. Um actor é um papel que um utilizador (ou outro sistema) assume na interacção com o sistema a modelar. O objectivo da modelação de casos de uso é identificar e

descrever as funcionalidades de alto nível que os actores pretendem do sistema. As descrições dos casos de uso são utilizadas para analisar e desenhar uma arquitectura do sistema que os possa implementar. Nesta metodologia, os diagramas de casos de uso do negócio constituem os modelos do negócio e os objectos de negócio modelizam-se com diagramas de classes.

Outro exemplo de modelação de negócio com UML é a metodologia proposta em 1999 por Chris Marshall [16], em que todos os aspectos de um negócio são modelados através de quatro conceitos relacionados entre si: o seu objectivo, processos, entidades e organização. Propõe a criação de dois tipos de modelos: modelos de objectivos e modelo de processos.

A metodologia de Eriksson-Penker [13], proposta em 2000, aproveitou o facto de a UML já ser uma linguagem madura, consolidada e amplamente utilizada na modelação de software e estendeu-a de modo a poder ser utilizada também na modelação de negócio. Como outras metodologias de modelação de negócio, o seu objectivo é produzir modelos do negócio para se acoplarem aos modelos de software. Contudo, Eriksson e Penker discordam da abordagem de metodologias como o RUP. Defendem que não é suficiente iniciar o desenvolvimento de um sistema pela modelação dos casos de uso do sistema de informação para definir os seus requisitos funcionais porque não há garantias que esses são os correctos e os que melhor suportam o negócio no qual o sistema opera. É necessário conhecer o sistema para examinar como os diferentes actores interagem, que actividades fazem parte do seu trabalho, quais são os seus principais objectivos, que regras os regem e mesmo em que aspectos podem ser melhorados.

A metodologia de Eriksson-Penker serviu de base à *Business Modeling Method* (BMM), proposta em 2004 por Montilva & Barrios [17]. Entre outras adaptações, incluiu-se a modelação das tecnologias que suportam o negócio por estes autores considerarem que podem condicionar o negócio.

Em 2001, a *Business Project Management Initiative* (BPMI) propôs a metodologia *Business Process Management* (BPM) [18] com objectivos mais ambiciosos: a descoberta, desenho, execução, interacção, operação e análise de processos de negócio. A intenção de produzir modelos de negócio que possam ser executados em linguagens levou ao desenvolvimento da notação *Business Process Modeling Notation* (BPMN) [27]. O seu propósito é ser uma notação gráfica utilizada para representar processos de uma forma que facilite a comunicação entre analistas, utilizadores e programadores e simultaneamente possa produzir modelos executáveis. Nesse sentido, foram desenvolvidas duas linguagens que executam processos de negócio: a *Business Process Modeling Language* (BPML) [28] que é uma metalinguagem baseada em XML para descrição de processos de negócio e que é interpretada por sistemas BPM (BPMS) e a *Business Process Execution Language* (BPEL) [29] que é uma linguagem de execução para a implementação de processos de negócio mediante a

composição de serviços *web*. Este tipo de metodologias serve de base a novas abordagens para a engenharia de software, como as designadas *model-driven development*. Constituem um passo adicional à aproximação entre a modelação de negócio e a modelação de sistemas de software em que se pretende que os modelos de negócio sejam directamente utilizados para orientar o desenvolvimento de software.

5.5 A.3 Conceitos de negócio

A metodologia de Eriksson-Penker identifica quatro conceitos fundamentais para representar um negócio: objectivos, processos, recursos e regras.

A.3.1 Objectivos de negócio

O objectivo primordial de qualquer organização é criar valor para os seus clientes. De uma forma geral, os objectivos derivam da missão da organização e traduzem o resultado final que se pretende atingir. Podem ser decompostos em sub-objectivos e alocados a partes individuais do negócio. Exprimem o estado pretendido dos recursos e são atingidos pelos processos. A sua definição pode ser apoiada por regras.

A.3.2 Processos de negócio

Os processos de negócio são actividades que produzem alterações ao estado dos recursos da organização. Um processo é caracterizado por utilizar recursos, ser composto por actividades, afectar uma ou mais estrutura da organização e criar valor para um grupo de clientes. Cada processo é responsável por coordenar e controlar as actividades que são executadas pelos recursos que actuam no negócio.

Um processo pode ser afectado por eventos que ocorrem no ambiente em que está envolvido ou gerados por outros processos. Os eventos de negócio são gatilhos que iniciam actividades ou que controlam as actividades a ser executadas. Vários eventos podem ocorrer durante a operação de um processo, aos quais este pode ter que reagir. Um processo pode também gerar eventos para outros processos dentro do negócio ou para outros negócios que irão causar a reacção destes de uma determinada forma.

A.3.3 Recursos

Os recursos são os objectos que actuam ou são utilizados no negócio. São consumidos, produzidos, transformados ou utilizados pelos processos de negócio. Alguns exemplos incluem material, energia, produtos, pessoas, informação e

serviços. Definem-se alguns estereótipos para indicar diferentes categorias de tipos de recursos. Na figura 32 apresenta-se um meta-modelo destes tipos.

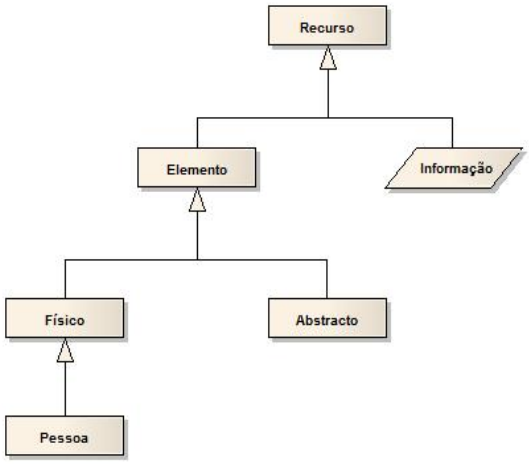

Figura 32 – Categorias de recursos de uma organização

Um recurso físico é uma entidade material que ocupa volume. Um recurso abstracto é uma ideia ou conceito. Um objecto de informação é a representação de um conceito. Contém informação sobre outros recursos e pode ser utilizado como seu substituto. É importante separar os objectos de informação dos próprios recursos que representam. As pessoas são uma especialização de recursos físicos criada para realçar a sua participação nos processos.

A.3.4 Regras de negócio

As regras de um negócio destinam-se a controlá-lo. Um modelo de negócio contém regras para definir restrições, condições e políticas sobre a forma como os processos de negócio devem ser executados. As regras de negócio podem afectar todos os outros conceitos, restringir a execução de um processo de negócio, o comportamento de um recurso e os meios para se atingir um objectivo específico. Restringem também as relações admissíveis entre conceitos. Podem ser definidas quer para satisfazer requisitos externos ao sistema (sob a forma de regulamentos, legislação ou restrições impostas por outros negócios) ou internamente para garantir que se atingem os objectivos do negócio. Definem-se três tipos de regras de negócio:

- Derivações – Regras que descrevem como é que o conhecimento numa forma pode ser trabalhado para gerar mais conhecimento. Pode ser uma regra computacional (fórmulas) ou uma regra de inferência (concluindo que a veracidade de um facto implica a veracidade de outro).

- Restrições – Regras que limitam a estrutura ou comportamento de objectos ou processos. Uma operação de um objecto pode ser restringida pela definição das pré-condições e das pós-condições, ou seja, condições que têm que ser satisfeitas antes e após a operação ser executada, respectivamente.
- Existência – Regras que definem quando um recurso pode existir ou ser criado e quando tem que ser eliminado.

As regras podem ser definidas formalmente numa linguagem específica ou em linguagem natural. Muitos diagramas da UML possuem suporte para definição de regras, usando construções genéricas para definir restrições. Um diagrama de classes tem restrições estruturais nas suas relações, como, por exemplo, a multiplicidade de uma associação. Um diagrama de estados tem restrições de comportamento nos seus estados e transições de estados (ex: acções a ser executadas quando ocorre uma transição). As regras de derivação podem ser definidas como restrições computacionais num diagrama UML (ex: o valor de um atributo pode ser calculado a partir do valor de outros atributos, como no caso da idade a partir da data de nascimento). Um diagrama de actividades impõe restrições de comportamento no próprio fluxo de actividades (ex: estruturas condicionais). A UML recomenda a *Object Constraint Language* (OCL) [30] como linguagem formal para especificar restrições, que é facilmente transferida para um sistema de informação que suporta o negócio e dessa forma tornar-se executável. A metodologia de Eriksson-Penker adopta também a OCL nos diversos modelos do negócio.

No âmbito das restrições, as regras de negócio que regem a forma como os conceitos se podem relacionar são as relações. Para além das relações entre estruturas dentro de cada categoria de recursos, existem também relações entre diferentes categorias de conceitos. Um processo de negócio está associado a um ou mais objectivos. Diferentes recursos são alocados a um processo de negócio, podendo cada ter os seus próprios objectivos. Podem haver regras que restringem os estados possíveis de um processo ou recurso. Os tipos de relações que podem ser encontrados nos diagramas UML, e consequentemente na metodologia de Eriksson-Penker, são:

- Transição de estados/actividades – Relação entre actividades e estados, mostrando que actividade ou estado se segue a outro. São chamadas de relações temporais, ou seja, mostram a sequência das actividades.
- Generalização – Relação de especialização/generalização em que objectos do elemento generalizado são substituíveis por objectos do elemento especializado. A generalização pode ser utilizada, por exemplo, para representar hierarquias entre diferentes recursos.
- Associação/agregação – Relação que descreve um conjunto de ligações entre vários objectos. A agregação é um caso especial de associação em que um

dos objectos contem ou controla outros objectos. Podem evidenciar relações de composição entre diferentes recursos.

- Dependência – Relação entre dois elementos em que, de alguma forma, um depende do outro. Tipicamente, pretende-se mostrar que uma alteração num objecto irá afectar o que dele depende. Uma dependência pode ser utilizada entre uma actividade e um objecto para mostrar que o objecto é consumido ou produzido pela actividade.

5.6 A.4 Perspectivas da modelação de negócio

A metodologia de Eriksson-Penker define quatro perspectivas diferentes para a modelação do negócio: visão do negócio, processos de negócio, estrutura do negócio e comportamento do negócio. Cada uma traduz uma forma diferente de observar o negócio, centrando a sua análise num conceito de negócio específico. Nem todas são utilizadas em todos os processos de modelação, contudo, a utilização de pelo menos duas torna a análise mais completa.

Cada perspectiva é representada por um ou mais diagramas. Os diagramas podem ser de diferentes tipos, dependendo da estrutura específica do negócio que está a ser modelizado. Os diagramas capturam e descrevem os conceitos de negócio, bem como as relações e interacções entre si.

A.4.1 Perspectiva da visão do negócio

A perspectiva da visão do negócio representa os objectivos da organização. Nesta perspectiva a metodologia inclui as seguintes técnicas:

- Definição estratégica – declaração da visão do negócio e análise estratégica, utilizando ferramentas como, por exemplo, matrizes SWOT[12];
- Modelação de objectivos e problemas – O modelo objectivo/problema é um diagrama de objectos da UML que decompõe os objectivos principais do negócio em sub-objectivos e indica os problemas que se interpõem para os atingir.
- Diagrama conceptual – diagrama de classes da UML que define os conceitos importantes e as suas relações no negócio para criar um conjunto comum de terminologia.

A.4.2 Perspectiva dos processos de negócio

A perspectiva dos processos do negócio representa as principais actividades realizadas na organização com vista à criação de valor. A metodologia de Eriksson-

[12] Uma matriz SWOT (*Strenghts, Weaknesses, Opportunities, Threats*) é uma ferramenta de análise de negócio que conjuga as oportunidades e ameaças do exterior com os pontos fortes e fracos de uma organização.

Penker recomenda que, para modelizar um processo, sejam colocadas as questões representadas na tabela 38, onde também se indicam os mecanismos de modelação que lhes podem dar resposta. A modelação de processos foca-se primeiro nos processos principais do negócio, passando depois para os processos de apoio.

Tabela 38 – Principais questões de modelação de processos

Questão de modelação	Mecanismo de modelação
Quais são as actividades necessárias?	Processos ou actividades no diagrama de processos
Quais são as actividades executadas e qual é o objectivo do processo?	Associação de um objectivo ao processo no diagrama de processo
Quando são executadas as actividades e por que ordem?	Ordem do fluxo de execução no diagrama de processo
Como são realizadas as actividades?	Decomposição dos processos em sub-processos que descrevem as actividades em maior detalhe no diagrama de processo
Quem ou o quê está envolvido na execução das actividades?	Recursos que participam no processo
O que é que está a ser consumido ou produzido no processo?	Entradas e saídas do processo
Como devem ser realizadas as actividades?	Fluxo de controlo ou pelas regras de negócio
Quem controla o processo?	Recursos que coordenam o processo

Eriksson e Penker utilizaram estereótipos para criar dois novos tipos de diagrama, ambos baseados em diagramas de actividades da UML: diagramas de processo e diagramas de linha de montagem.

Diagramas de processo

Os diagramas de processo são diagramas de actividades da UML que, utilizando estereótipos, descrevem as actividades executadas dentro de um processo, a forma como interagem entre si e os recursos que nelas participam. A figura 33 representa um diagrama de processo genérico.

Os objectos que representam recursos que estão envolvidos no processo são colocados à volta do processo:

- Objectivo – Objecto que representa a motivação para se realizar o processo. Relaciona-se com este através de uma dependência estereotipada «atingir», indicando que o processo pretende atingir o objectivo indicado. É representado por cima do processo.
- Controladores – Recursos que controlam ou executam o processo relacionando-se com este através de uma dependência estereotipada «controlar». São representados por cima do processo.
- Entradas – Recursos consumidos ou refinados pelo processo. São representados à esquerda do processo.

- Saídas – Recursos produzidos pelo processo ou que são resultado do refinamento de um ou mais objectos de entrada. São representados à direita do processo.
- Fornecedores – Recursos envolvidos no processo mas que não são consumidos nem refinados, relacionando-se com este através de uma dependência estereotipada «fornecer».

O diagrama de processo por incluir também mais dois tipos de elementos:

- Eventos – Objectos que representam ocorrência de factos. Podem ser recebidos pelo processo e este reagir a ele de alguma forma (por exemplo, despoletando a sua execução) ou ser gerados pelo processo, de forma a ser captados por outro processo.
- Restrições – Descrições textuais ou expressões OCL que especificam condições (*pre-conditions* à entrada e *post-conditions* à saída do processo) e que têm que se verificar para o diagrama ser válido.

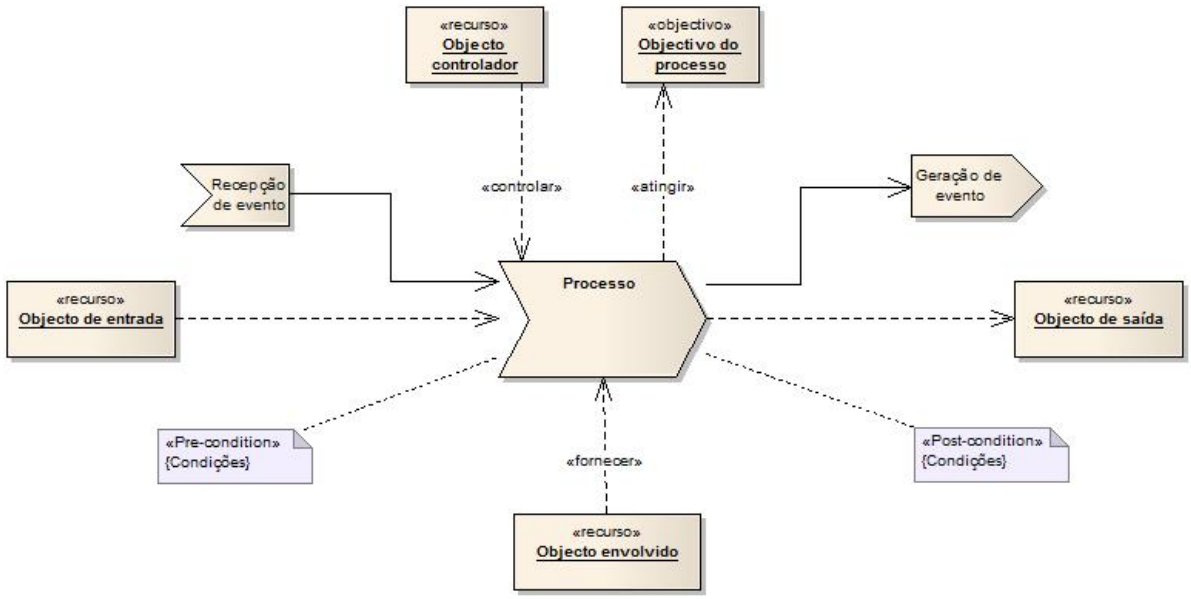

Figura 33 – Diagrama de processo genérico

Diagramas de linha de montagem

Os diagramas de linha de montagem são diagramas de actividades da UML que, com um conjunto de estereótipos, descrevem a interacção entre vários processos e classes de recursos na cadeia de valor do negócio. A figura 34 representa um digrama de linha de montagem genérico. Segundo Eriksson e Penker, este diagrama tem particular interesse para modelação de processos, quando o objectivo é o desenvolvimento de um sistema de informação, pela forma como representa as interfaces entre processos e recursos. Na secção superior coloca-se um diagrama de

processo. Na secção inferior, um diagrama de pacotes representando uma linha de montagem com uma sucessão de objectos de um tipo de recurso. Cada processo consulta informação de um objecto ou adiciona-lhe nova informação. O propósito do diagrama é demonstrar de que forma vários processos interagem com os mesmos recursos. Frequentemente, os objectos nos pacotes de linha de montagem mapeiam-se em recursos informativos num sistema de informação.

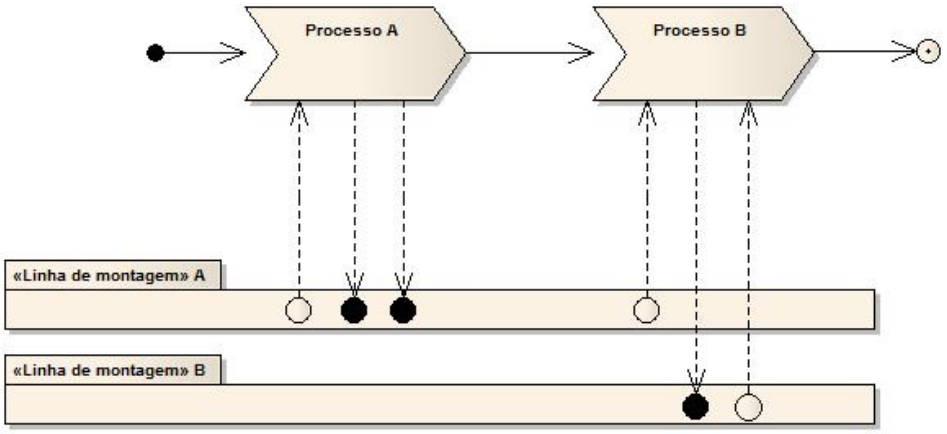

Figura 34 – Diagrama de linha de montagem genérico

A.4.3 Perspectiva da estrutura do negócio

A perspectiva da estrutura do negócio representa as estruturas dos recursos, dos produtos ou dos serviços e a informação no negócio, incluindo a estrutura organizativa. São baseados em mapas de estruturação de conceitos normalmente já existentes na organização. A informação da perspectiva dos processos também é aproveitada uma vez que mostra quais são os recursos utilizados. Note-se que, tipicamente, estas perspectivas são modelizadas em paralelo, uma vez que contribuem uma para a outra e têm que se consistentes. Esta vista mostra as estruturas dos recursos, da informação e da estrutura organizacional do negócio.

À semelhança da perspectiva da visão do negócio, os diagramas elaborados são diagramas conceptuais. Contudo, focalizam-se mais na estrutura dos recursos, da informação produzida e da organização.

A.4.4 Perspectiva do comportamento do negócio

A perspectiva do comportamento do negócio representa o comportamento individual dos seus recursos e processos bem como a sua interacção. O comportamento dos objectos de recurso é regido pela perspectiva dos processos de negócio, que mostra o principal fluxo de como o trabalho é efectuado. A perspectiva do comportamento do negócio analisa cada um dos objectos envolvidos

em maior nível de detalhe: o seu estado, o seu comportamento em cada um e as possíveis transições de estado. Analisa também as interacções entre diferentes processos e como se sincronizam uns com os outros.

Os diagramas produzidos são diagramas de sequência e diagramas de colaboração. Os diagramas de processo podem também ser utilizados para mostrar a interacção entre processos.

5.7 A.5 Da modelação do negócio à modelação do sistema de software

No contexto da especificação de um sistema de software, a modelação do negócio é a forma para determinar se os casos de uso definidos para um sistema são os mais correctos ou os mais indicados para esse sistema. Existem outras utilizações para um modelo de negócio, já que muitos dos objectos e relações nele encontradas serão também objectos e relações no modelo do sistema de informação. É importante, no entanto, reconhecer que não se trata de um mapeamento directo. O modelo de negócio carece sempre de uma análise crítica para averiguar que partes são aplicáveis para um sistema de informação específico.

Segundo Eriksson e Penker, o modelo do negócio é utilizado no desenvolvimento de modelos de software para identificar os sistemas de informação que melhor suportam a operacionalização do negócio (como novos sistemas, sistemas no mercado ou sistemas legados). Os modelos de negócio são utilizados como base para identificar o conjunto correcto de funções ou casos de uso que o sistema deve oferecer aos processos de negócio. Desta forma contribuem para a descoberta dos requisitos funcionais.

Embora não seja possível transferir directamente as classes do modelo de negócio para o modelo do software, a informação sobre um recurso no modelo de negócio pode ser utilizado para identificar classes no sistema.

Anexo B

Metodologia de modelação de software com casos de uso

5.8 B.1 Introdução

A modelação de software com casos de uso é uma técnica que assenta no pressuposto que para se compreender o que um sistema deve fazer é necessário investigar como se pretende que seja utilizado e com que objectivos. Teve as suas origens nos meados dos anos 80 quando Jacobson [19] apresentou a ideia de cenários de utilização de sistemas de software.

O propósito da modelação de software por casos de uso é descrever de forma sucinta e acessível o comportamento de um sistema na perspectiva de uma "caixa preta", ou seja, apenas em termos do que é percepcionado a partir do exterior. O conjunto de casos de uso de um sistema descreve todas as formas com que este pode ser utilizado e o comportamento que apresenta em cada uma.

Segundo Gorman [20], a metodologia de modelação de casos de uso de um sistema pode ser utilizada para capturar requisitos funcionais, planear e gerir o projecto, orientar a concepção de testes do sistema, apoiar a produção de documentação para o utilizador e mesmo para descrições de negócio.

Nas secções seguintes apresenta-se um resumo da metodologia e dos principais mecanismos de modelação que propõe.

5.9 B.2 Conceitos

A metodologia de modelação de software com casos de uso identifica três conceitos fundamentais: actores, casos de uso e cenários de utilização.

B.2.1 Actores

Os actores são as entidades externas que interagem com o sistema. Podem ser pessoas, máquinas ou outros sistemas. Os actores são caracterizados por um nome, uma descrição curta e pela indicação dos casos de uso em que estão envolvidos. A designação "actor" advém do facto de o utilizador desempenhar um papel em relação ao sistema. Por definição, os actores são entidades sempre fora do sistema. Cockburn [31] propõe a distinção entre actores primários e secundários. Na sua perspectiva, os actores primários são os seus utilizadores, ou seja, que interagem com o sistema com a intenção de atingir um objectivo específico. Os actores secundários são os que auxiliam o sistema para que este possa satisfazer os objectivos dos actores primários.

B.2.2 Casos de uso

Os casos de uso representam os objectivos que um ou mais actores pretendem do sistema. Descrevem formas dos actores interagirem com o sistema e como este responde às suas acções para cumprir o objectivo pretendido. São o conceito central da técnica de modelação de sistemas de software por casos de uso. Segundo Bittner e Spence [22], os casos de uso obrigam a focar a análise do sistema no valor que este possui para os actores, em contraste com a especificação de um conjunto de funções arbitrárias que podem não satisfazer directamente os objectivo de um actor. Fomenta também o confronto com a usabilidade do sistema devido à ênfase na interacção. Os casos de uso são caracterizados por um nome, uma descrição e um conjunto de elementos opcionais.

B.2.3 Cenários de utilização

Quando um actor interage com o sistema, no âmbito de um caso de uso, efectua um conjunto de diferentes acções que correspondem a um fluxo de eventos. Esse fluxo é uma abstracção de cenários de utilização do sistema. Um cenário é uma instância de um caso de uso e representa um percurso específico na sua realização [21]. Pode construir-se um cenário a partir do fluxo genérico de um caso de uso e outros cenários a partir de variações desse fluxo. Embora o resultado final possa ser diferente em cada cenário, dependendo das circunstâncias, todos eles se relacionam com o mesmo objectivo funcional e tem o mesmo ponto de início. De facto, a mais-valia da modelação de um sistema com casos de uso encontra-se na descrição dos seus cenários de utilização.

5.10 B.3 Descoberta de casos de uso

Vários autores apresentam métodos semelhantes para descobrir casos de uso. De uma forma geral é recomendada a realização dos passos de descoberta dos actores, casos de uso e descrição dos respectivos fluxos de eventos, como apresentado na tabela 39.

Tabela 39 – Metodologia genérica de descoberta de casos de uso

Passo	Descrição
1. Descobrir os actores	Identificar todos os possíveis utilizadores do sistema. Determinar os papéis que esses utilizadores desempenharão relativamente à sua interacção com o sistema. Criar um actor para cada papel. Alguns incluem: administrador, utilizador para consulta, utilizador para uma área funcional específica.
2. Descobrir casos de uso	Criar um caso de uso para cada objectivo, mantendo o mesmo nível de abstracção.
3. Descrever o fluxo de eventos	Uma vez identificado um caso de uso, os actores com que interage e o seu objectivo, é possível efectuar o desenho de interacção a alto nível. Esse desenho é descrito sob a forma de fluxos de eventos que ocorrem da interacção de cada actor com o sistema.

Na prática, não é simples separar os três passos, uma vez que são interdependentes. A descoberta de actores, por vezes, faz-se à custa de se encontrarem casos de uso e vice-versa. Explorar os requisitos contidos nos casos de uso pode levar à identificação de mais actores que podem, por sua vez, levar à necessidade de mais casos de uso. Deve, contudo, garantir-se que todos os actores participam em, pelo menos, um caso de uso. Os que não o fizerem são considerados supérfluos ou denunciam que ainda há casos de uso por descobrir.

Segundo Gorman [20], um erro frequente ao descrever fluxos é a indicação de detalhes técnicos e a associação de desenhos de interface específicos ou de tecnologias de implementação. Nesta descrição deve capturar-se a lógica das interacções e dar particular ênfase ao seu propósito. Por esse motivo, devem utilizar-se frases como "utilizador confirma a sua escolha" em vez de "o utilizador pressiona o botão *Enter*".

Enquanto se identificam os casos de uso, podem encontrar-se requisitos que não se enquadram facilmente neste modelo. Estes devem ser capturados como parte de uma especificação suplementar que deve ser criada em paralelo com o modelo de casos de uso. O *Rational Unified Process* [1] prevê explicitamente a existência de um documento suplementar de requisitos que complemente o documento formal.

5.11 B.4 Relações

Para além das associações entre actores e casos de uso, os próprios casos de uso podem associar-se entre si. As relações entre casos de uso são utilizadas para evidenciar o que possuem em comum e, simultaneamente, para reduzir a sua

complexidade. Um comportamento partilhado por vários casos de uso pode ser abstraído para um caso de uso próprio, promovendo a sua melhor legibilidade e consistência da descrição. De igual forma, um comportamento que se aplica apenas num contexto muito específico de um caso de uso pode ser isolado noutro, melhorando a compreensão do inicial. Para suportar este tipo de situações utilizam-se as relações de inclusão, extensão e generalização entre casos de uso.

B.4.1 Relações de inclusão

As relações de inclusão permitem extrair secções comuns de descrições de um ou mais casos de uso e colocá-las em casos de uso separados de onde podem ser referenciados. Para que possam ser utilizadas é necessário ter a mesma descrição de comportamento em pelo menos dois casos de uso. Portanto as relações de inclusão só devem ser incluídas depois de se ter escrito as descrições, de forma a detectar repetições. Ao caso de uso base é acrescentado, num ponto específico, o comportamento do caso de uso incluído. Uma característica do caso de uso incluído é a ausência de referências aos casos de uso que o incluem. Tal implica que casos de uso incluídos são reutilizáveis porque não estão restritos a nenhum contexto em particular. Segundo Bittner e Spence [22], o erro mais comum ao utilizar a relação de inclusão é usá-la para efectuar a decomposição funcional de um sistema. Um exemplo dessa abordagem é tratar o caso de uso incluído como uma opção de um menu ou uma função. Os casos de uso incluídos tendem a não ter comportamento comum e serem incluídos apenas num caso de uso. A consequência dessa opção de modelação é que nenhum desses casos de uso por si só apresenta valor para os actores, só quando combinados uns com os outros.

B.4.2 Relações de extensão

As relações de extensão são utilizadas em situações em que se pretende evidenciar um comportamento opcional ou excepcional num caso de uso. A característica do caso de uso de extensão é que não implica alterações ao caso de uso que estende. Tal implica que o caso de uso base (também designado neste contexto de caso de uso estendido) tem que permanecer inalterado, ser completo e continuar a fazer sentido mesmo sem a extensão, para continuar a ter valor. As circunstâncias que podem constituir extensões incluem descrições de características opcionais, tratamento de erros, excepções ou outras que interfiram com a descrição do comportamento geral do sistema. Por vezes também se utilizam extensões para descrever comportamento que só será introduzido em versões posteriores. Ao contrário do caso de uso incluído, o caso de uso de extensão depende do caso de uso base que estende. O caso de uso base é estendido em um ou mais pontos de extensão ou sob determinadas condições.

B.4.3 Relações de generalização

A relação de generalização permite criar descrições de comportamentos genéricos que podem ser especializados para responder a necessidades específicas. Os actores podem associar-se tanto ao caso de uso generalizado como aos casos de uso que são especializações desse. A especialização possui os mecanismos da inclusão porque reutiliza a descrição noutro caso de uso (o caso de uso generalizado), mas com a semântica de uma extensão, uma vez que especializa o caso de uso com comportamento adicional. Um caso de uso pode ser especializado em qualquer número de pontos de extensão e pode ser estendido em mais do que um caso de uso de extensão. O caso de uso base não tem que estar completo e com significado, pode até ter pontos em branco que só são completados com a especialização.

Os actores de um sistema também podem relacionar-se. Contudo, a única relação possível é a generalização que é utilizada para evidenciar similaridades entre os actores, ou seja, a partilha de responsabilidades ou características. Não é utilizada para reflectir permissões de segurança. Estas devem ser asseguradas pelo sistema e descritas nos casos de uso. Não existe a necessidade de outras relações entre actores porque estes não comunicam directamente entre si no contexto do sistema. No modelo de casos de uso do sistema, ignoram-se completamente todas as interacções que possam ocorrer entre os actores fora do sistema.

5.12 B.5 Representação e descrição de casos de uso

B.5.1 Diagramas de casos de uso

Um diagrama de casos de uso é um tipo de diagrama de modelação do comportamento de um sistema de software definido na *Unified Modeling Language* (UML) [21]. O seu propósito é apresentar numa representação esquemática os actores, os casos de uso e as relações entre si, como representado na figura 35. Note-se que o diagrama de casos de uso por si só não descreve um sistema. As descrições encontram-se nos textos que lhes estão associados. O diagrama de casos de uso serve como uma panorâmica ou um sumário do comportamento do sistema.

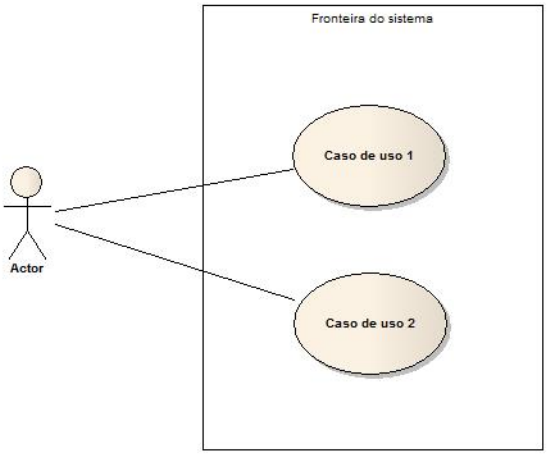

Figura 35 – Diagrama de casos de uso genérico

A especificação formal da UML [8] define a notação gráfica para modelizar casos de uso com diagramas. Porém, não define nenhum formato para a sua documentação e estruturação de descrição. Embora a notação gráfica seja importante, deve sempre ser acompanhada da descrição, ou seja, dum conjunto de elementos textuais que expliquem o caso de uso. Nas secções seguintes indicam-se esses elementos e os seus objectivos na descrição.

B.5.2 Nome e descrição sumária

O nome tem que ser único em todo o modelo e deve ser da forma «verbo»+«substantivo». O verbo deve ser conjugado no modo do infinitivo e indicar a acção a ser realizada com o caso de uso.

A descrição sumária é um pequeno texto que indica claramente o objectivo que o(s) actor(es) pretende(m) atingir ao realizar o caso de uso. O nome do caso de uso e a sua descrição sumária constituem a informação mínima que se deve incluir na sua documentação.

B.5.3 Fluxos de eventos

Um fluxo de eventos descreve interacções entre um ou mais actores e o sistema. Os cenários de utilização são concretizações particulares de fluxos de eventos que ilustrem casos reais de combinações de eventos.

A descrição de fluxos de eventos deve iniciar-se com a descrição do fluxo básico, que corresponde ao cenário principal. Traduz o que deve acontecer quando o actor consegue realizar o caso de uso, atingindo o seu objectivo sem a ocorrência de percalços, excepções ou situações alternativas. Para a descrição dos fluxos de eventos identificam-se dois estilos de descrição:

- Estilo coloquial – A descrição é estruturada numa forma de diálogo em que um actor gera eventos e o sistema responde com acções simples, podendo

ser representado através de uma tabela. É adequada em casos de uso em que só exista um actor e a natureza do cenário seja inerentemente interactiva, ou seja, quando os eventos alternam sempre entre o actor e o sistema. Não é recomendada quando existe mais do que um actor ou quando a resposta do sistema é complexa.

- Estilo narrativo – A descrição é estruturada em passos numerados sequencialmente em que a acção vai oscilando entre vários actores e o sistema. Este formato é mais flexível porque permite que o sistema também possa iniciar acções e suporta a interacção de múltiplos actores. Parece também ser o mais adequado a acomodar alterações aos casos de uso pois é fácil introduzir novos passos na sequência. Pode ser representado através de uma tabela. Contudo, convém sempre clarificar se os eventos partem do sistema ou do actor.

Segundo Bittner e Spence [22], uma descrição do fluxo de eventos não deve descrever só o que acontece na interface do sistema (por exemplo, um ecrã). Embora um caso de uso se centre na perspectiva do exterior, ou seja, a do utilizador relativamente ao sistema, é admissível que também inclua o comportamento interno deste. Se ocorrem validações, mesmo que o utilizador não se aperceba delas, devem ser incluídas.

Após a descrição do fluxo básico devem identificar-se os fluxos opcionais, alternativos e excepcionais. São os fluxos que correspondem aos cenários de utilização diferentes do principal. Os fluxos opcionais implicam que ocorre comportamento opcional, mas que o actor pode decidir não fazer. Os fluxos alternativos dizem respeito a escolhas que o actor tem de fazer. Uma das alternativas (a mais frequente) está no cenário principal, enquanto outras, vão para os cenários alternativos. Os fluxos de excepção implicam a ocorrência de situações muito pouco frequentes, normalmente associadas à ocorrência de anomalias.

Este tipo de fluxos podem ser registados na descrição do fluxo básico se os passos alternativos forem poucos e curtos e não perturbarem a compreensão desse fluxo. Contudo, se o número de passos alternativos aumentar, a inclusão no fluxo básico pode perturbar a sua compreensão. Nessas situações, são remetidos para descrições autónomas. Devem ter um nome que descreva o seu objectivo e ser numerados sequencialmente. Essa numeração fornece uma referencia que é útil quando se discutem vários fluxos alternativos ou quando são apresentados em outros documentos.

B.5.4 Pontos de extensão

No fluxo de eventos básico, se for utilizado o estilo narrativo, podem associar-se nomes a alguns passos. Estes nomes designam-se de pontos de extensão porque

permitem que o fluxo de eventos possa ser estendido no passo em que ocorrem. Se se mover o comportamento alternativo, opcional ou de excepção para uma secção autónoma, é necessária uma forma de referenciar o ponto em que o comportamento é inserido ou substituído pelo básico.

Normalmente, os fluxos diferentes do fluxo básico são realizados num ou mais pontos de extensão deste. Em algumas situações, no entanto, podem não ocorrer em nenhum ponto de extensão específico, mas apenas quando se verifica uma condição. Um exemplo é a ocorrência de uma falha geral. No que diz respeito ao que ocorre na sua conclusão, configuram-se três situações possíveis:

- Indicar que retoma a execução do caso de uso no ponto de extensão em que tinha sido interrompido, implicando que introduz comportamento adicional ao fluxo básico, normalmente associado a fluxos opcionais.
- Indicar que termina o caso de uso, implicando o que é autónomo do básico a partir do ponto de extensão, normalmente associado a fluxos de excepção.
- Indicar o ponto de extensão em que deve ser retomado o fluxo básico, implicando constituir uma variante ao fluxo básico, normalmente associada a fluxos alternativos.

É possível que os fluxos diferentes do fluxo básico tenham eles próprios fluxos opcionais, alternativos ou de excepção. A técnica a utilizar é a mesma, ou seja, definir pontos de extensão no próprio fluxo para referenciação e depois indicar o que deve acontecer quando este termina.

B.5.5 Pré-condições e pós-condições

As pré-condições descrevem o estado em que o sistema tem que estar para o caso de uso poder ser iniciado. Esse estado é descrito pela verificação da conjunção das pré-condições. Exemplos de pré-condições são a disponibilidade de um recurso ou a existência de informação de contexto.

As pós-condições descrevem o estado em que o sistema tem que ficar após o caso de uso ser concluído. Esse estado é descrito pela verificação de pelo menos uma disjunção das pós-condições. A indicação de pós-condições só é necessária quando esse estado é importante para um dos actores do caso de uso, ou seja, quando esse estado está relacionado com o objectivo pretendido.

Frequentemente as pós-condições de um caso de uso são pré-condições de outro. As pré-condições podem ser encaradas como requisitos que tem que ser verificados antes do caso de uso ser realizado. Se é antecipável que a indicação de pós-condições facilitará o planeamento de testes, também é um motivo para que devam ser incluídas. Várias pós-condições, por defeito, actuam em disjunção umas relativamente às outras. Tanto as pré-condições como as pós-condições devem ser escritas em frases simples.

5.13 B.6 Recomendações

A modelação de casos de uso é uma actividade que exige alguma experiência em modelação. Na tabela 40 resumem-se algumas recomendações sugeridas por Bittner e Spence [22] para orientar as descrições de casos de uso, nomeadamente para equipas que estejam a introduzir pela primeira vez a modelação com casos de uso em processos de engenharia de requisitos.

Tabela 40 – Recomendações para as descrições de casos de uso

Recomendação	Descrição
Escrever de forma simples e directa	No texto das descrições e fluxos de eventos, utilizar a voz activa e discurso directo, escrever frases no presente e não no futuro e evitar verbos que introduzam ambiguidade como "deve".
Tratar o caso de uso como uma história	A descrição do caso de uso deve ser uma narração de uma utilização do sistema. O início deve ser sempre um evento despoletado por um actor e o final deve ser bem explícito.
Decidir o nível de detalhe na descrição	O nível de detalhe na descrição de cada caso de uso está directamente relacionado com o estado em que deve ser documentado. Decidir qual é esse estado permite focalizar o esforço da descrição do caso de uso e compreender quando está concluído.
Clarificar a descrição com outros elementos	Se necessário, deve complementar-se a descrição textual com diagramas, esquemas ou tabelas que promovam a clarificação do que está a ser explicado.

Anexo C

Análise de modelos de documentos de especificação de requisitos

5.14 C.1 Introdução

Qualquer organização que pretenda produzir um documento de especificação de requisitos tem duas opções para a sua estruturação: adoptar um modelo já existente ou criar o seu próprio modelo. Os inconvenientes da primeira opção são as vantagens da segunda. Na primeira opção, embora possa ser cómodo dispor de um modelo já completamente delineado em que todos os capítulos e secções estão estruturados, nem sempre essa estruturação se enquadra com os conteúdos necessários para cada projecto em particular ou com os impostos pelo cliente. Na segunda opção, mais trabalhosa porque implica derivar um modelo, é possível adaptar cada documento de especificação de requisitos a cada projecto em particular. Na realidade, os próprios autores dos modelos já existentes reiteram a segunda opção, ou seja, as organizações podem basear-se nos modelos mas devem sempre tentar alterá-los para que melhor acomodem as particularidades de cada projecto.

Nesse sentido, para obter uma perspectiva que permita compreender os conteúdos típicos de um documento de especificação de requisitos, foram analisados três modelos já existentes:

- Modelo do IEEE [11] – Modelo proposto na norma IEEE 830-1998 como recomendação de uma forma de organizar um documento de especificação de requisitos;
- Modelo do RUP [25] – Modelo baseado no do IEEE, proposto na metodologia *Rational Unified Process* (RUP) [1] mas com alterações substanciais na forma de descrever o sistema e enquadrar os requisitos em casos de uso;

- Modelo de Withall [24] – Modelo proposto por Withall para a escrita de documentos de especificação de requisitos.

Na tabela 41 apresenta-se a estrutura de cada modelo. Como se verifica, há coincidência de conteúdos em alguns capítulos e diferenças em outros. Tal deve-se ao facto de cada modelo partir de abordagens diferentes para a descrição do sistema e também da forma como organiza os próprios requisitos.

Tabela 41 – Modelos de documentos de especificação de requisitos

Modelo do IEEE	Modelo do RUP	Modelo de Withall
1. Introdução	**1. Introdução**	**1. Introdução**
1.1. Objectivo	1.1. Objectivo	1.1. Objectivo do sistema
1.2. Âmbito	1.2. Âmbito	1.2. Propósito do documento
1.3. Definições, acrónimos e abreviaturas	1.3. Definições, acrónimos e abreviaturas	1.3. Formato das definições dos requisitos
1.4. Referências	1.4. Referências	1.4. Glossário
1.5. Visão geral	1.5. Visão geral	1.5. Referências
		1.6. Versões do documento
2. Descrição geral	**2. Descrição geral**	**2. Contexto**
2.1. Perspectiva do produto	2.1. Modelo genérico de casos de uso	2.1. Âmbito
2.2. Funções do produto	2.2. Pressupostos e dependências	2.2. Principais pressupostos
2.3. Características dos utilizadores		2.3. Principais exclusões
2.4. Restrições		2.4. Principais entidades de negócio
2.5. Pressupostos e dependências		2.5. Infra-estruturas
3. Requisitos específicos	**3. Requisitos específicos**	**3. Áreas funcionais**
3.1. Requisitos de interfaces com o exterior	3.1. Relatório de casos de uso	…
3.2. Requisitos funcionais	3.2. Requisitos suplementares	
3.3. Requisitos de desempenho		
3.4. Requisitos de bases de dados		
3.5. Restrições de desenho		
3.6. Atributos do sistema de software		
Anexos	**Anexos**	**4. Principais características não-funcionais**

Nas secções seguintes exploram-se os conteúdos de cada modelo assinalando as semelhanças e comparando as diferenças. Pretende-se com esta análise apoiar a concepção da estrutura de um documento de especificação de requisitos específico para uma organização e/ou projecto.

5.15 C.2 Capítulo de introdução

Nas secções seguintes apresentam-se os tópicos incluídos pelos modelos analisados no capítulo "Introdução" que inicia o documento de especificação de requisitos.

C.2.1 Introdução do documento

Em qualquer dos modelos, o documento inicia-se com o capítulo "Introdução". Os modelos do IEEE e RUP propõem a mesma estrutura para este capítulo enquanto o de Withall, embora cobrindo os mesmos tópicos, apresenta uma estrutura diferente.

Outros aspectos a ter em atenção nesta secção são os seguintes:

- Público-alvo – é recomendável identificar todos os grupos de pessoas para as quais o documento se destina e assegurar que o estilo de escrita e conteúdos do documento são adequados a todas as pessoas que o integram;
- Esclarecimentos – os leitores devem ser alertados que se uma característica não está explicitamente especificada, deve assumir-se que não será incluída no sistema;
- Estrutura do documento – deve descrever-se a forma como o documento está organizado;
- Outras especificações de requisitos – deve indicar-se se há outros requisitos que o sistema tem de satisfazer e que estão definidos em outros documentos de especificação.

Os modelos do IEEE e do RUP concluem o capítulo com uma visão geral do mesmo que inclui uma descrição do conteúdo dos capítulos seguintes. Withall opta por incluir essa informação na secção "Propósito do documento".

C.2.2 Objectivos do documento

Os modelos do IEEE e RUP apresentam os objectivos do documento na secção "Objectivo" enquanto no de Withall tal é designada "Propósito do documento". Independentemente do nome que lhe seja atribuído, pretende-se que nesta secção seja explicado o motivo pelo qual o documento foi escrito, esclarecendo-se que a especificação de requisitos de software descreve o comportamento externo do sistema e os requisitos não-funcionais, restrições de desenho e outros aspectos para uma descrição completa e detalhada dos requisitos de software. Deve também ser indicado qual o é público-alvo a que o documento se destina, para esclarecimento do leitor.

C.2.3 Objectivos do sistema

Nos modelos do IEEE e RUP, os objectivos do sistema são descritos na secção "Âmbito". Segundo a norma do IEEE deve fazer-se uma breve descrição do sistema ao qual a especificação de requisitos se refere, incluindo características e enquadramento com outros sistemas, modelo de casos de uso associado e outros aspectos que são afectados ou influenciados pelo documento. No modelo do RUP

recomenda-se que se identifiquem os produtos de software, explicando o que esses sistemas devem fazer e, se necessário o que não farão.

No modelo de Withall, estes conteúdos surgem na secção "Objectivo do sistema" com a descrição do motivo que levou à criação do sistema, quem são os interessados e os seus principais objectivos. O tamanho desta secção varia em função da complexidade e natureza do sistema. Withall prossegue ainda com uma secção especificamente destinada à apresentação do formato dos requisitos que foi adoptado para o documento.

C.2.4 Glossário

Os modelos do IEEE e do RUP incluem uma secção para definições, acrónimos e abreviaturas. Nela devem incluir-se o indispensável para interpretar correctamente a terminologia do documento. Segundo a norma do IEEE, esta informação pode também ser indicada por referência ao glossário do projecto, que se recomenda incluir como anexo.

No modelo de Withall a secção é designada "Glossário", definindo-a como um conjunto de definições de termos utilizados no documento. Segundo Withall, um glossário deve ser sempre incluído e produzido à medida que os conceitos vão sendo descritos, com os seguintes objectivos:

- Definir o significado de termos relevantes – as definições dos termos no glossário são consideradas oficiais, implicando que a sua utilização no texto do documento tenha que ser explicitada na definição;
- Instruir os leitores não informados – a leitura sequencial do glossário pode informar uma pessoa introduzida recentemente no projecto sobre a terminologia, permitindo que possa discuti-lo com o resto da equipa;
- Combater ideias pré-concebidas – ao ser alvo de uma aprovação oficial, o glossário torna-se a referência para as definições, desfazendo outras ideias que possam coexistir sobre os termos;
- Impor a definição de conceitos de domínio mal compreendidos – se um termo é difícil de definir (ou se os membros da equipa não chegam a acordo sobre o mesmo), tal é sintoma de que não está bem compreendido.

O formato de um glossário é uma lista de termos e respectivas definições. Os termos devem ser distintos e sem ambiguidades, listados alfabeticamente e sem repetições. Se houver termos com sinónimos, o termo será indicado uma única vez, com as definições para cada sinónimo.

C.2.5 Referências

Os três modelos incluem uma secção com as referências do documento. De acordo com o modelo do IEEE e RUP as referências são uma lista com todos os documentos utilizados ou citados no documento de requisitos. Withall menciona que as referências identificam as fontes de informação utilizadas durante a escrita do documento. Existem vários formatos possíveis e válidos para apresentação de referências.

C.2.6 Histórico de versões

O modelo de Withall é o único que indica explicitamente uma secção para o controlo de versões do documento. Nela lista todas as versões do documento, com a informação de nº de versão, data, autor que criou o documento na primeira versão e/ou autores que efectuaram alterações, lista resumida de alterações efectuadas. Embora os outros modelos não a refiram explicitamente, a informação sobre as versões é tipicamente incluída no início de qualquer documento técnico.

5.16 C.3 Capítulo de descrição geral ou contexto

Nas secções seguintes apresentam-se os tópicos incluídos pelos modelos analisados no capítulo seguinte ao da introdução. Normalmente neste capítulo faculta-se informação genérica e de suporte que permita contextualizar o sistema. Embora atribuindo nomes diferentes, "Descrição geral" nos modelos do IEEE e RUP e "Contexto" no de Withall, o intuito do capítulo é o mesmo, porém, a forma utilizada para organizar e apresentar essa informação é substancialmente diferente.

C.3.1 Descrição geral do sistema

No modelo do IEEE o capítulo de descrição geral inicia-se com a exposição dos factores gerais que afectam o produto e os seus requisitos. Note-se que o modelo não se refere a requisitos específicos já que estes são descritos no capítulo seguinte. Em vez disso, apresenta a informação de apoio de forma a serem mais facilmente perceptíveis. Inclui secções como "Perspectiva do produto", "Funções do produto", "Características dos utilizadores", "Restrições" e "Pressupostos e dependências".

A secção "Perspectiva do produto" destina-se a colocar produto junto de outros com que está relacionado. Se o produto a desenvolver constituir um componente de um sistema maior, deve apresentar-se a arquitectura geral, com os principais componentes, interconexões, interfaces externos e os requisitos gerais desse sistema. Descrevem-se também as várias interfaces que o produto terá com outros sistemas, com o sistema em que se insere, com utilizadores, com interfaces com hardware, com interfaces de software (interfaces com outros produtos) e outros interfaces de

comunicação em geral. Na secção "Funções do produto" resumem-se as principais funcionalidades que o sistema disponibilizará. Na secção "Restrições" apresentam-se as descrições de quaisquer outros itens que não tenham sido contemplados nas secções anteriores e que possam auxiliar a limitar as opções da equipa de desenvolvimento. Alguns exemplos destes incluem políticas de auditoria, segurança, protecção de dados e plataformas de desenvolvimento. Estão associadas normalmente a requisitos não-funcionais.

O modelo do RUP está enquadrado na metodologia de modelação de software com casos de uso. Nesse sentido, a perspectiva geral do sistema é apresentada pela visão geral de casos de uso. Na secção "Modelo genérico de casos de uso" ilustra-se de forma gráfica, através de um diagrama de casos de uso ou de pacotes, o conjunto de casos de uso que o sistema oferece aos seus actores.

No modelo de Withall o capítulo "Contexto" inicia-se com uma secção "Âmbito" com um diagrama de contexto, seguido de notas explicativas. O diagrama de contexto referido por Withall não adere a nenhuma notação específica (não é um diagrama UML). Tem como propósito representar o sistema no ambiente que o rodeia, descrevendo os participantes (actores no equivalente em UML) e estabelecendo as fronteiras do sistema. O diagrama de contexto representa o sistema e as entidades (componentes, pessoas, fronteira do âmbito, interfaces, ligações, armazenamento de dados e agrupamentos) com que terá que lidar. Apresenta uma perspectiva lógica do sistema para que qualquer leitor a possa interpretar. O diagrama é acompanhado de uma breve descrição de cada componente. Dessa forma, todo o público-alvo do documento fica com uma perspectiva da complexidade do sistema. Procura-se que todos os componentes sejam apresentados da mesma forma (os visíveis da interface e os menos visíveis), condensando-se toda a informação num único diagrama.

C.3.2 Descrição de entidades e actores

O modelo do IEEE inclui uma secção de "Características dos utilizadores". Nela descrevem-se os prováveis utilizadores do sistema, por nível de formação, experiencia e competências técnicas. No modelo do RUP, os utilizadores são apresentados na secção "Modelo genérico de casos de uso" em que surgem modelizados como actores do sistema. Withall indica que as entidades de negócio sejam descritas na secção "Principais entidades de negócio". Na sua interpretação, as entidades incluem todos os objectos, classes e conceitos que rodeiam o sistema e com as quais este terá que lidar. Para cada uma, sempre que pertinente, deve ser apresentado o seu ciclo de vida.

C.3.3 Pressupostos, exclusões e dependências

Tanto o modelo do IEEE como o modelo do RUP incluem uma secção de "Pressupostos e dependências" para a descrição da viabilidade técnica, disponibilidade de componentes ou outras condições que tem que se verificar para o sistema pode ser operacionalizado. Por exemplo, a norma do IEEE indica como pressuposto que um sistema operativo específico tenha que estar instalado no hardware onde o produto será colocado em produção.

O tópico é mais detalhado no modelo de Withall, que o separa em três secções: "Principais pressupostos", "Principais exclusões" e "Infra-estruturas".

Na secção "Principais pressupostos", indicam-se os aspectos que devem ser assumidos para o sistema poder ser operacionalizado. Podem ser utilizados para indicar dependências externas, ou seja, factores que estão fora do controlo do sistema mas nos quais este depende. É importante referir os pressupostos para que o público-alvo do documento se possa pronunciar, e se necessário, identificar situações em que estes possam não se verificar. Withall recomenda que os pressupostos que só digam respeito a uma área funcional do sistema sejam remetidas para o capítulo respectivo. O formato dos pressupostos é uma lista com três itens: identificador, declaração e justificação.

Além de indicar os pressupostos, o modelo de Withall inclui uma secção de "Principais exclusões", para identificar os principais aspectos que o sistema não suportará. Devem ser indicadas as exclusões para as características que poderiam ser esperadas pelo público-alvo. Tal permite antecipar algumas dúvidas dos leitores possam ter sobre o sistema. Note-se que, embora exclusões não sejam o mesmo que pressupostos, os dois conceitos estão relacionados. Uma funcionalidade do sistema pode estar excluída porque se assume que já está presente noutro sistema. Nessas situações, será suficiente indicá-la como pressuposto, não sendo necessário indicá-la também como exclusão. O formato das exclusões é idêntico ao dos pressupostos. O modelo do RUP, na secção "Objectivos do sistema" do capítulo "Introdução", faz uma menção mais sucinta à possível indicação de exclusões.

O modelo de Withall isola numa secção de "infra-estruturas" os pressupostos que estão directamente ligados à existência de componentes. Esses componentes, embora não façam parte do sistema, são necessários para que possa ser operacionalizado, alinhando-se com o conceito de pressupostos apresentado pelos modelos do IEEE e RUP.

5.17 C.4 Capítulo de especificação de requisitos

Nos três modelos analisados, o capítulo seguinte ao da descrição geral ou contexto consiste na sua descrição concreta dos requisitos em diferentes níveis de detalhe. A forma como o capítulo é organizado varia bastante pois cada modelo

adopta uma abordagem diferente: no modelo do IEEE os requisitos são agrupados em categorias; no modelo do RUP os requisitos são enquadrados em casos de uso e suplementares; no modelo de Withall os requisitos são enquadrados em áreas funcionais.

C.4.1 Organização de requisitos por categorias

Segundo a norma do IEEE, esta é a parte mais importante e volumosa de todo o documento devendo conter todos requisitos de software num nível de detalhe que permita que tanto os desenhadores como os testadores possam garantir que o sistema desenvolvido os cumpre. No modelo do IEEE os requisitos são organizados em diferentes secções por categorias, como indicado na tabela 42.

Tabela 42 – Categorias de requisitos de acordo com a norma IEEE 830-1998

Categoria de requisitos	Descrição
Interfaces com o exterior	Descrição de todas as interfaces do sistema com utilizadores, hardware e software de entrada e saída de dados. As interfaces assumem normalmente a forma de formulários, relatório e mensagens. Para cada uma indicam-se objectivo, origem e destino, formato e *layout* e detalhes de cada item.
Funcionais	Descrição de todas as acções fundamentais que ocorrem no sistema quando recebe entradas e gera saídas de dados. Inclui, por exemplo, verificações de validação de dados, sequência exacta de operações realizadas, resposta a situações anómalas, recuperação de erros, fórmulas de cálculo, etc.
Desempenho	Descrição de medidas de desempenho do software declaradas de forma quantitativa. Inclui, por exemplo, métricas sobre o número de utilizadores suportados em simultâneo, número de terminais que podem estar a interagir com o sistema em simultâneo e a quantidade e tipo de informação que tem que ser capaz de lidar, etc.
Bases de dados	Descrição de aspectos relacionados com o armazenamento persistente de informação no sistema. Inclui entidades de dados e as suas relações, restrições de integridade, concorrência, privilégios de acesso, frequência de utilização e políticas de retenção.
Desenho	Descrição de algumas restrições sobre a forma como o sistema deve ser desenhado, por exemplo, para que adira a normas ou imposições legais.
Atributos do sistema de software	Descrição de atributos que o produto deve exibir, nomeadamente, relacionados com fiabilidade, disponibilidade, segurança, manutenção e portabilidade.

C.4.2 Organização de requisitos por casos de uso

No modelo do RUP, os requisitos são enquadrados em casos de uso e aqueles em que tal não for possível são considerados requisitos suplementares. Em conformidade com esta distinção, o modelo separa os requisitos para a secção de "Relatórios de casos de uso" e para a secção "Requisitos suplementares".

O relatório de casos de uso consiste no detalhe dos casos de uso incluído na secção "Modelo genérico de casos de uso" do capítulo "Descrição geral". Não

existindo uma norma para as descrições de casos de uso, pode ser adoptada uma que inclua a informação prevista pela metodologia de modelação de casos de uso.

Os requisitos suplementares incluem todos aqueles que não é possível enquadrar em casos de uso. Tal pode incluir requisitos funcionais transversais a todo o sistema ou requisitos não-funcionais. O modelo indica que, opcionalmente, estes requisitos podem ser referenciados em outros documentos de especificação.

C.4.3 Organização de requisitos por áreas funcionais

No modelo de Withall os requisitos são organizados por áreas funcionais. Nesse sentido, propõe que seja efectuada a decomposição funcional do sistema por grupos lógicos de funcionalidades. Ao contrário dos outros modelos, Withall defende que deve ser criado um capítulo próprio para especificar os requisitos de cada grupo resultante da decomposição funcional. A forma de criar os grupos lógicos pode basear-se nos principais processos de negócio ou se tal não for facilmente determinado, a partir do que é realizado por grupos de utilizadores. A ordem dos capítulos deve reflectir a importância de cada área funcional. Devem ser descritas primeiro as que forem consideradas de maior importância. Os critérios para definir a importância podem estar relacionados com a frequência com que as funções são realizadas ou do nível de interesse que os utilizadores lhes atribuem. Tipicamente, a área funcional mais directamente relacionada com os objectivos do negócio é considerada a mais importante.

Independentemente da forma como se descobrem os grupos de funcionalidades, o modelo de Withall prevê que existam requisitos funcionais que não são facilmente atribuíveis a nenhum. Nesse caso, devem ser incluídos num capítulo adicional para os acomodar, nomeado, por exemplo, "Outros tópicos". Note-se que para descrever os requisitos não-funcionais o modelo prevê o capítulo "Principais características não-funcionais".

5.18 C.5 Capítulo de especificação suplementar

Os modelos analisados encerram o documento com aquilo que consideram ser especificação suplementar. Contudo, a própria definição de especificação suplementar difere entre os modelos.

Nos modelos do IEEE e RUP, como todos os requisitos são incluídos no capítulo "Especificação de requisitos", a restante informação que se pretende incluir é remetida para anexos. Estes podem conter, por exemplo, protótipos de interface para o utilizador, relatos de utilização derivados de casos de uso. Quando os anexos são incluídos deve ser explicitado se fazem parte dos requisitos ou se constituem apenas informação adicional de contexto.

No modelo de Withall, considera-se especificação suplementar todos os requisitos tipicamente classificados de requisitos não-funcionais e que são transversais a todo o sistema. Por esse motivo, incluem um último capítulo designado de "Principais características não-funcionais". O próprio autor do modelo recomenda que o capítulo seja o mais pequeno possível. Se houver necessidade de também incluir requisitos funcionais, que sejam transversais ao sistema, o capítulo deve ser renomeado para "Outras capacidades do sistema".