FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

FEUP

Gestão de conteúdos localizados dos produtos clínicos ALERT®

João Manuel Bonita Pereira Loureiro

Relatório de Projecto/Dissertação

Mestrado Integrado em Engenharia Informática e Computação

Orientador: Gabriel de Sousa Torcato David (Prof. Associado da FEUP)

Responsável de acompanhamento: Rui Spratley (Eng.)

3 de Março de 2009

Gestão de conteúdos localizados dos produtos clínicos ALERT®

João Manuel Bonita Pereira Loureiro

Relatório de Projecto/Dissertação

Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo júri:

Presidente: Jorge Alves da Silva (Prof. Auxiliar da FEUP)

Arguente: José Maria Amaral Fernandes (Prof. Auxiliar Convidado da Universidade de Aveiro)

Vogal: Gabriel de Sousa Torcato David (Prof. Associado da FEUP)

20 de Março de 2009

Confidencial

Nos termos do protocolo de estágio e do acordo de confidencialidade celebrado com a ALERT Life Sciences Computing, S.A. (”ALERT”), o presente relatório é confidencial e poderá conter referências a invenções, know-how, desenhos, programas de computador, segredos comerciais, produtos, fórmulas, métodos, planos, especificações, projectos, dados ou obras abrangidos por direitos de propriedade industrial e/ou intelectual da ALERT. Este relatório só poderá ser utilizado para efeitos de investigação e de ensino. Qualquer outro tipo de utilização esta sujeita a autorização prévia e por escrito da ALERT.

In accordance with the terms of the internship protocol and the confidentiality agreement executed with ALERT Life Sciences Computing, S.A. (“ALERT”), this report is confidential and may contain references to inventions, know-how, drawings, computer software, trade secrets, products, formulas, methods, plans, specifications, projects, data or works protected by ALERT’s industrial and/or intellectual property rights. This report may be used solely for research and educational purposes. Any other kind of use requires prior written consent from ALERT.

Resumo

Com o advento da globalização, o recurso a mercados estrangeiros, mais do que uma fonte de rendimento extraordinária, constitui condição necessária para o crescimento sustentado das empresas. Para as empresas de *software* em particular, desenvolver produtos preparados para diferentes mercados traduz-se na necessidade de adoptar metodologias específicas denominadas internacionalização e localização, nem sempre compreendidas por todos os intervenientes.

A ALERT Life Sciences Computing, S.A. é uma empresa na área da saúde que rapidamente ganhou presença no mercado internacional, contando já com uma vasta distribuição dos seus produtos pelo mundo. No contexto deste cenário, determinou que seriam necessárias melhorias ao nível do processo de localização dos seus produtos por forma a não comprometer a sua expansão. É essa a motivação do projecto de que trata este relatório.

Os processos de internacionalização e localização de *software* são entendidos como simbióticos, sendo que o primeiro, de natureza mais estruturante, influencia directamente o segundo, de natureza mais processual. Ao analisarmos a forma como a indústria suporta estes processos, verificamos ainda que diferentes decisões tecnológicas direccionam para diferentes formas de implementação. Particularmente no que concerne à localização, concluímos que a definição de um processo, ferramentas e tecnologias apropriados são essenciais à sua eficácia.

Neste sentido, foi efectuada uma análise técnica em detalhe da *suite* de produtos ALERT® no que diz respeito à sua internacionalização. A arquitectura do ALERT® assenta numa base de dados relacional na qual é persistida a totalidade dos seus conteúdos localizados. O processo de desenvolvimento de *software* tem influência na forma como esses conteúdos são produzidos, traduzidos e incluídos no ALERT®, ou seja, influencia o processo de localização. A dificuldade em coordenar os dois processos afecta negativamente a eficiente produção do *software* para os diferentes mercados-alvo .

Como forma de resolver muitos dos problemas identificados, optou-se por centralizar o armazenamento do conteúdo localizado num repositório para o efeito. Adjacente a este repositório, propõe-se uma aplicação Web, à qual se ligam os colaboradores do departamento de desenvolvimento e os tradutores, com o objectivo de suportar o processo de tradução dos conteúdos localizados. Este suporte passa por fornecer aos utilizadores funcionalidades que permitam acompanhar o processo desde a criação dos conteúdos localizados, passando pela sua tradução e terminando na sua extracção para efeitos de versionamento. A solução é apresentada recorrendo fundamentalmente à especificação da sua estrutura estática, da qual se extrai o modelo de dados relacional do repositório. É esperada uma implementação com base nesta especificação e a integração da ferramenta daí resultante nos processos da empresa.

Abstract

With the advent of globalization, the resort to foreign markets constitutes a necessary condition for sustained business growth, rather than a mere source of extraordinary income. For software companies in particular, developing products ready for use in different markets means adopting specific methodologies named internationalization and localization, which are often not clearly perceived by all stakeholders.

ALERT Life Sciences Computing, SA is a health IT company that quickly made its way to international markets and today its products are distributed all over the world. In this context, ALERT determined that its software localization process needed improvement so as not to compromise its expansion. This is the motivation behind the project documented by this report.

The processes of software internationalization and localization are regarded as symbiotic. The former, more structuring by nature, directly influences the latter, which is more procedural. By analyzing how the industry supports these processes, it was found that different technological decisions lead to different implementations. In what localization is concerned, in particular, we also concluded that appropriate choices regarding processes, tools and technologies are essential to its effectiveness.

The ALERT® suite of products was then the subject of a detailed technical analysis with regard to its internationalization. The architecture of ALERT® lies on a relational database in which all of its localized content is persisted. The software development process influences how this content is created, translated and included in ALERT® — that is to say, it influences the localization process. Difficulty in coordinating the two processes hinders the efficient production of the software for the different target markets.

In order to solve many of the problems identified, it was chosen to centralize the storage of the localized content in a repository for this purpose. A Web application sitting on top of the repository, which both the software development team and the translators connect to and which provides support for the translation of the localized content, is proposed. Features that allow users to follow this process from the content creation, through its translation, to its extraction for versioning, provide that support. The proposed solution is presented essentially by means of a specification of the static structure of the system, from which the relational data model of the repository is drawn. An implementation of the specification is expected by the company, as well as the integration of the resulting tool within its processes.

Agradecimentos

Prof. Gabriel David, Eng. Rui Spratley, André Tavares, Área *Citizen*, em particular Rui Neves, Luís Nóbrega e Paulo Ferreira, João Eiras e Nuno Guerreiro.

J. L.

Conteúdo

Lista de Figuras

Abreviaturas e Símbolos

API	Application Programming Interface
Cap.	Capítulo
cf.	Confira, confronte
CSV	Comma Separated Values
DDL	Data Definition Language
DLL	Dynamic-link library
DML	Data Manipulation Language
HTML	HyperText Markup Language
ISO	International Organization for Standardization
Java SE	Java Platform, Standard Edition
OASIS	Organization for the Advancement of Structured Information Standards
PL/SQL	Procedural Language/Structured Query Language
Sec.	Secção
SVN	Subversion
UML	Unified Modeling Language
VPN	Virtual Private Network
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language

Capítulo 1

Introdução

1.1 Enquadramento

Ainda que a popularidade do termo "globalização" verificada a partir da década de 1990 se deva sobretudo à sua acepção como fenómeno social, com significado nem sempre preciso, é provavelmente a sua vertente económica aquela que é mais tangível. Com esse sentido, designa o processo de integração gradual de mercados distintos que, teoricamente, conduz à emergência de um mercado único à escala mundial. Podemos considerar que esta globalização económica se inicia tão remotamente quanto as explorações de Portugal e Espanha nos séculos XV e XVI, mas num contexto mais actual identificamo-la como resultado da prossecução dos princípios do neoliberalismo por parte das entidades políticas.

Na prática, esta globalização significa que as empresas já não competem somente a nível nacional. Muitas das vezes, o seu crescimento está mesmo dependente do aumento da percentagem das receitas além-fronteiras. As empresas de *software* não são excepção, pelo contrário. Acompanhando a generalidade do sector de alta-tecnologia, o mercado do *software* tem-se tornado cada vez mais globalizado nos últimos anos [Int02]. Esta mudança de paradigma é acompanhada pela mudança de expectativas dos consumidores, que progressivamente apresentam mais resistência à ideia de uma língua franca nos produtos de *software*, como o foi o inglês durante décadas.

As receitas adicionais vêm, portanto, a um preço. Mais do que assegurar o correcto esquema mental por parte de quem desenvolve, produzir *software* internacional estende-se a toda a organização. Com efeito, a falta de envolvimento por parte da gestão de topo pode resultar numa subestimação do esforço e trabalho requeridos para criar uma mesma aplicação preparada para ser utilizada em mercados diferentes. A ausência de um planeamento pode mesmo acabar por tornar esses esforços contraproducentes.

Como em qualquer outro processo, tempo e dinheiro são palavras-chave. Idealmente, queremos minimizar o custo da internacionalização relativamente ao desenvolvimento de um produto nacional. Ao mesmo tempo, queremos que daí resulte um produto muito flexível e também fácil de localizar. Ao nível da localização, desejamos a maior eficiência possível, ajustando correctamente o nível de esforço e despesa ao potencial retorno. Atingir este cenário é um desafio que envolve tanto questões técnicas como questões estratégicas. E se já são os próprios requisitos de negócio actuais que impedem, à partida, que se pense e idealize em termos monoculturais, o desafio não se esgota aí:

> " 'It works' is not enough. 'Now that's a brilliant idea that will change the way people do things' is a much more fetching proposition."[Int02]

Este relatório versa sobre um projecto proposto pela empresa ALERT Life Sciences Computing, S.A. (ALERT) no âmbito da localização - nomeadamente da tradução - da sua *suite* principal de produtos.

1.2 Empresa

A ALERT, com sede em Vila Nova de Gaia, Portugal, é a empresa-mãe de um grupo de empresas denominado Grupo de Empresas ALERT. Este grupo encontra-se distribuído por outros pontos do mundo, nomeadamente E.U.A., Espanha, Holanda, Singapura, Brasil e Reino Unido. Tem por missão:

> "Melhorar a saúde e prolongar a vida, alcançar rentabilidade para benefício da sociedade e inspirar outros para a excelência, através do nosso exemplo."

O Grupo de Empresas ALERT dedica-se ao desenvolvimento, distribuição e implementação do *software* de saúde ALERT®, concebido para criar ambientes clínicos sem papel [ALE08a]. Todo o desenvolvimento é realizado em Portugal, tendo as subsidiárias responsabilidades ao nível da distribuição, comercialização e instalação dos produtos nos respectivos mercados.

O ALERT® é actualmente distribuído em 31 países e está implementado em mais de 600 hospitais e cerca de 8000 centros de saúde. Encontra-se disponível em 6 línguas — inglês, espanhol, italiano, holandês, português europeu e brasileiro e francês. Está prevista a disponibilização do ALERT® em alemão, eslovaco, croata, chinês, russo, japonês e árabe a curto prazo.

Na Figura 1.2 podemos ver um diagrama da família de produtos ALERT®. A ALERT oferece soluções integradas cobrindo os diversos ambientes clínicos. Assim, o ALERT® RHIO (Regional Health Information Organization) designa a solução completa ao nível de cuidados de saúde para uma região ou país que, por sua vez, se subdivide nas soluções

Figura 1.1: Distribuição ALERT® pelo mundo.

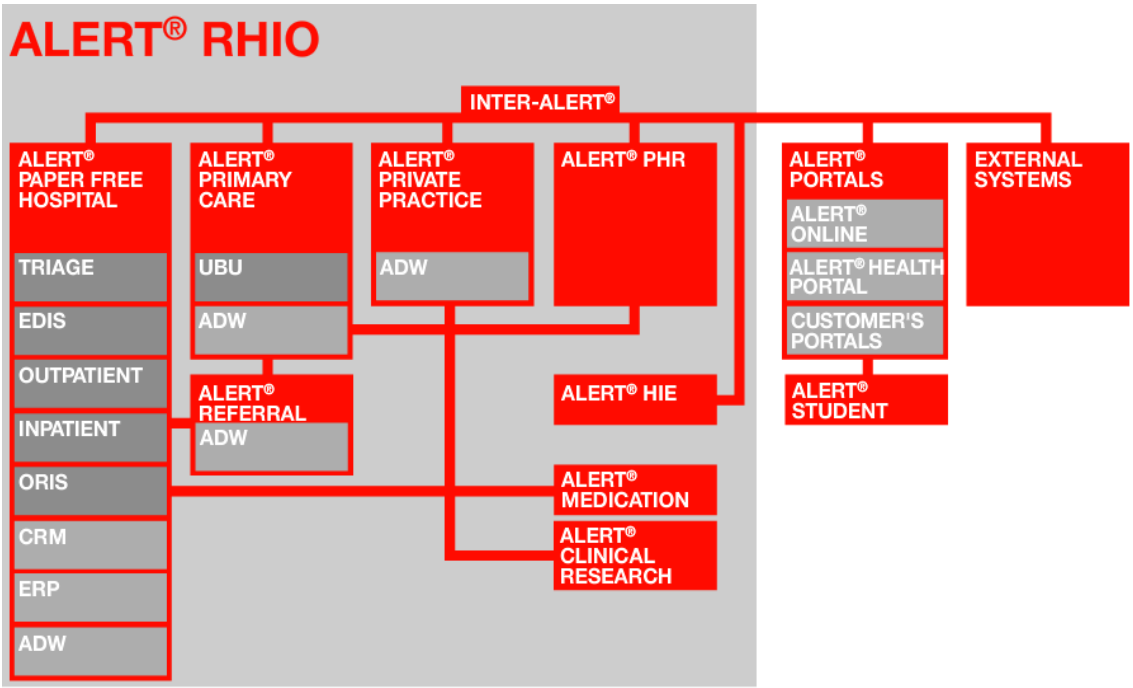

Figura 1.2: Diagrama da família de produtos ALERT®.

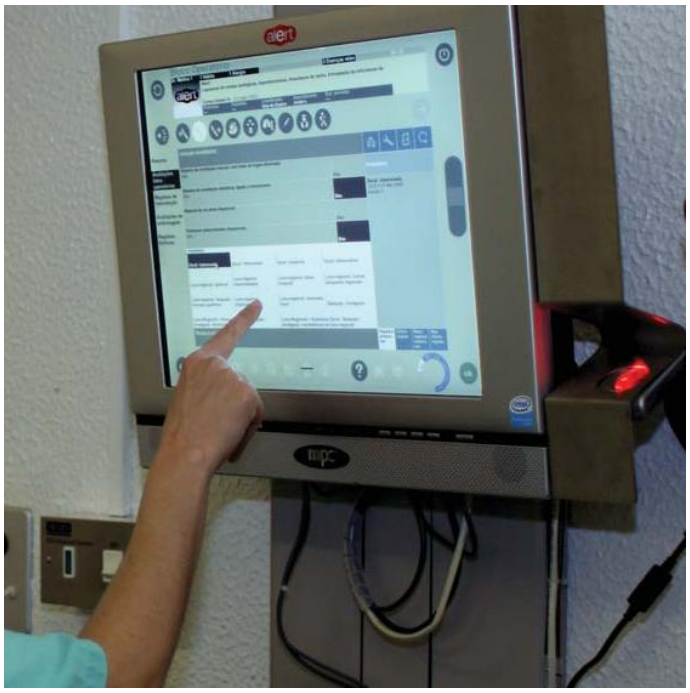

Figura 1.3: Profissional usando o ALERT®.

para hospitais (ALERT® PAPER FREE HOSPITAL), centros de saúde (ALERT® PRIMARY CARE) e clínicas privadas (ALERT® PRIVATE PRACTICE). Ao nível dos hospitais, em particular, existem também produtos específicos para cada um dos ambientes: triagem (ALERT® TRIAGE), serviço de urgência (ALERT® EDIS), consulta externa (ALERT® OUTPATIENT), internamento (ALERT® INPATIENT) e bloco operatório (ALERT® ORIS). Existem também soluções não-clínicas (ALERT® CRM e ERP) e de *datawarehousing* (ALERT® ADW).

Como algumas características distintivas, o ALERT® apresenta uma interface do tipo *touchscreen*, perfis de utilização diferenciados para os diferentes profissionais de saúde, estabelecimento de fluxos de trabalho entre os mesmos e identificação biométrica.

A ALERT tem conseguido manter um crescimento sustentado apostando progressivamente no mercado global. Em 2008, os negócios nos mercados externos representaram cerca de 60% do volume de negócios da empresa. Recentemente, a ALERT firmou novos contratos no Brasil e Reino Unido, assistiu ao primeiro lançamento do EDIS nos E.U.A. e Malásia e reforçou a sua distribuição no Médio Oriente.

1.3 Projecto

1.3.1 Motivação

O presente projecto surge da identificação de possíveis melhorias ao nível da gestão dos conteúdos localizados dos produtos por parte da própria empresa, para a qual estes se revestem de elevada importância. Citando a proposta oficial ([FEU08]):

- "Na abordagem a um novo mercado, a possibilidade de utilização dos produtos clínicos ALERT® no idioma nativo é sempre um factor decisivo para adopção do software."
- "A informação dependente do mercado e da localização geográfica existente no ALERT® encontra-se actualmente colocada em cada uma das tabelas da base de dados, não existindo um repositório na verdadeira acepção da palavra, o que dificulta os processos de actualização de dados e upgrades de versão."
- "Após este desenvolvimento, as implementações, tanto a nível nacional como internacional do produto, serão simplificadas, sendo vital para a vertente de internacionalização da empresa (...)"

Concomitantemente às questões de foro mais técnico, são também identificados problemas e dificuldades recorrentes que atravessam o próprio processo de localização, que se pretende mais ágil. Uma resenha destes problemas pode ser encontrada em [Eir08], como iremos ver no Cap. 3.

1.3.2 Objectivos

Importa começar por referir que o projecto não consistiu na "reformulação do modelo de dados e código do ALERT®, no que toca ao armazenamento e obtenção de conteúdos localizados", como constava de [FEU08], tendo a empresa optado por manter, pelo menos por ora, as decisões de arquitectura a esse nível. Isto implica que o âmbito do projecto se desvia da internacionalização do produto, focando-se na localização.

Recorrendo novamente à proposta de projecto, daí podemos extrair os seguintes objectivos:

- "Análise técnica e desenvolvimento, recorrendo a tecnologias de base de dados ORACLE e PL/SQL, de um repositório centralizado para gestão de conteúdos dependentes do mercado e da localização geográfica."
- "(...) mecanismos de importação e exportação de conteúdos, necessários para a integração e actualização de conteúdos localizados em ambientes de desenvolvimento, controlo de qualidade e produtivos."

- "(...) eliminação e substituição de dependências do restante modelo de dados (mais de 1000 tabelas) e, consequentemente, pela implementação de uma entidade independente que disponibiliza a informação através de uma API criada para o efeito."
- "O sistema deverá ser capaz de importar conteúdos localizados a partir de uma qualquer instalação do software ALERT®, independentemente da sua versão, de forma parcial ou total."
- "O sistema permitirá a exportação de conteúdo localizado a partir do repositório para uma qualquer base de dados ORACLE de destino."
- "O sistema possibilitará a obtenção de conteúdo por traduzir e a actualização do mesmo."

1.4 Estrutura do Relatório

Para além da introdução, este relatório contém mais 4 capítulos. No capítulo 2, é descrito o estado da arte. No capítulo 3, é feita uma descrição mais detalhada do problema. No capítulo 4, é apresentada a solução proposta. No capítulo 5, são apresentadas as conclusões a tirar desta fase do projecto bem como direcções relativamente ao trabalho futuro.

Capítulo 2

Estado da Arte

2.1 Internacionalização e Localização

Internacionalização e localização são termos usados na indústria e na literatura de *software* para designar duas problemáticas inerentes ao desenvolvimento de *software* internacional [1]. Entendemos por *software* internacional um mesmo produto preparado para ser utilizado em mercados geográfica e culturalmente díspares [Int02]. Obter um produto nessas condições implica endereçar num conjunto de requisitos que não correspondem, necessariamente, a requisitos inerentes ao domínio de aplicação do produto [SSJtET02]:

- Requisitos linguísticos;
- Questões culturais;
- Diferenças políticas;
- Considerações financeiras;
- Factores geográficos.

O facto de encontrarmos invariavelmente estes conceitos juntos deve-se não só ao facto de visarem o mesmo desafio, como também por designarem processos simbióticos, na medida em que apenas quando aplicados em conjunto resultam eficazmente no objectivo de produzir *software* internacional. São, contudo, processos distintos e que ocorrem em fases diferentes do ciclo de desenvolvimento do produto.

De uma maneira geral, internacionalização refere-se às questões de desenho e codificação que permitem que um produto seja flexível relativamente a especificações de

[1] Em inglês, *internationalization* e *localization*, respectivamente, e com frequência informalmente abreviados para i18n e l10n devido ao número de letras que medeia entre a letra inicial e final.

diferentes mercados [Sun01]. Como tal, é um processo de generalização [Fou08]. Segue-se uma ilustração simples mas paradigmática de internacionalização no que concerne à apresentação de mensagens ao utilizador. Confronta aquele que poderá ser o aspecto de uma base de código não internacionalizada relativamente a uma internacionalizada [Sun01]:

```
printf("This message needs internationalization.");

printf(catgets(mycatalog, 1, 1, "This message is now
    internationalized."));
```

Enquanto que no primeiro exemplo a mensagem é introduzida directamente no código (*hard-coded*), no segundo é externalizada recorrendo a um catálogo, permitindo que o conteúdo que é enviado para o ecrã seja determinado em tempo de execução. Note-se que, no segundo exemplo, interessam os identificadores numéricos e não a *string*, que apenas é usada caso a função não consiga carregar a mensagem. Localização, por outro lado, consiste no processo de adaptação de um produto a um determinado mercado e ocorre tipicamente depois da sua internacionalização (um dos papéis da internacionalização é tornar o produto localizável [Sun01, chap. 2.4]). O processo de localização pode compor-se de várias actividades: tradução, da interface com o utilizador, por exemplo; formatação de determinados campos com base em convenções culturais, como datas, horas e números; entre outras — de uma forma geral, as actividades de adaptação direccionadas aos requisitos internacionais em causa (cf. requisitos levantados anteriormente).

Se é verdade que estes conceitos estão relativamente bem estabelecidos, é de referir que a terminologia usada nem sempre é consistente entre referências. Por exemplo, [Int02] considera internacionalização como sendo a actividade de criar *software* internacional no seu todo, sendo esta composta pelos processos de *world-readiness* e localização — onde por *world-readiness* se entende o que se aqui definiu como sendo internacionalização. Os próprios autores referem:

> " 'World-ready,' 'international,' 'international-aware,' 'internationalized,' 'globalized,' and 'worldwide' are all buzzwords describing programs that are designed to function for more than one language."

Neste relatório, os termos internacionalização e localização têm as acepções descritas anteriormente.

2.2 Técnicas de internacionalização

Se a extensão da internacionalização de um produto é determinada por decisões estratégicas e requisitos de negócio, o modo como esta se consegue ao nível da implementação é muitas das vezes influenciada, ou mesmo determinada, por decisões de arquitectura

antecedentes, como a escolha da plataforma tecnológica. De facto, actualmente a maioria das plataformas de programação traz consigo suporte ao nível das API (*National Language Support*), de ferramentas e de práticas estabelecidas no que à internacionalização das suas aplicações diz respeito. Considerem-se alguns casos.

No caso de aplicações Microsoft, sejam aplicações Win32 ou .NET, o método de implementação consistirá em separar o conteúdo localizável do código-fonte da aplicação, tipicamente em ficheiros do tipo *resource* - Microsoft Win32 *resource files* tradicionais ou .resx para aplicações .NET. O binário compilado conterá apenas o cerne da aplicação abstraindo os aspectos linguísticos. Juntamente, são incluídas DLLs, preferencialmente uma por língua, correspondentes a esses ficheiros *resource*. Isto permite efectivamente ter uma interface multilingue, uma vez que bastará carregar a DLL correspondente à língua pretendida, que poderá ser a da interface do sistema operativo, por exemplo, ou poderá ser dada ao utilizador essa escolha [Int02]. Neste contexto, conteúdo localizável significará mais precisamente conteúdo com necessidade de tradução que, como podemos inferir a partir do exposto anteriormente, é um subconjunto do conteúdo localizável de uma aplicação.

Em aplicações Web, os princípios seguidos são os mesmos, sendo no entanto mais comum armazenar o conteúdo localizável em ficheiros em XML. Assumindo que é mantida a mesma informação entre os diferentes ficheiros XML correspondentes às diferentes línguas, aplicamos a mesma transformação XSL para HTML em tempo de visualização.

Em aplicações nas quais toda a persistência de dados é conseguida recorrendo a uma base de dados relacional, isto é, não só dados ditos "transaccionais" como também de "parametrização" são guardados em base de dados, é provável que os recursos localizáveis sejam também persistidos e acedidos dessa forma. Nesse caso, é necessário internacionalizar o modelo de dados que suporta a aplicação. Na Figura 2.1 são apresentadas três alternativas propostas por [SSJtET02] relativamente ao desenho de um modelo de dados internacionalizado. O exemplo dado é o de um catálogo de itens de produtos. Cada item tem associada uma descrição textual.

A alternativa recomendada é a terceira (mais à direita) por ser a mais flexível relativamente à manipulação de informação localizada. Ao colocar-se essa informação numa tabela de detalhes, cuja chave é composta por uma referência à tabela principal (ITEMID) e pelo *locale*, a adição de nova informação significa apenas uma inserção na tabela e não representa modificações ao nível do código. Resumidamente, *locale* é um identificador linguístico e geográfico. A aplicação acede aos dados por identificador do item e por *locale*. Mais uma vez, o paradigma assenta em não ter informação *hard-coded* ao nível do código aplicacional, neste caso a informação relativa ao *locale*.

2.2.1 Locales

Um *locale* é definido em [Int02] da seguinte forma:

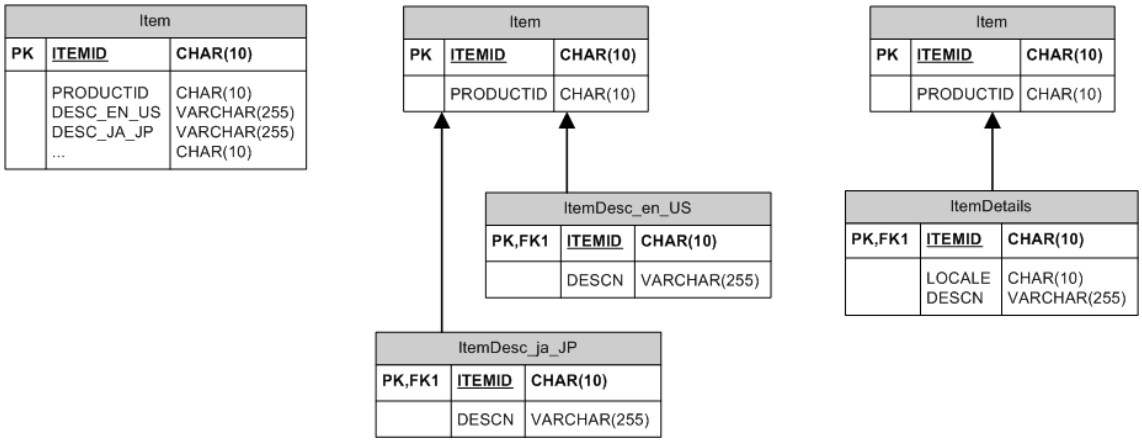

Figura 2.1: Três alternativas de modelação de uma base de dados internacionalizada.

> "The collection of features of the user's environment that is dependent on language, country/region, and cultural conventions. The locale determines conventions such as sort order; keyboard layout; and date, time, number, and currency formats."

É usual vermos um *locale* representado pelos códigos ISO para línguas e países (normas ISO 639-1 e ISO 3166-1, respectivamente) tal como recomendado por [Alv01] para identificadores de línguas (*language tags*). Usando essa convenção, "en-GB" representa o *locale* "Inglês (Reino Unido)", por exemplo. Relativamente a esse *locale*, o ambiente de uma aplicação apresentaria a interface em inglês britânico e pressuporia uma série de outras convenções culturais existentes no Reino Unido. A definição de *locale* apresenta ligeiras variações de plataforma para plataforma, mas regra geral todas partem do formato recomendado por [Alv01] para identificadores de línguas. A API de Java SE, como exemplo, define assim Locale [Sun08]:

```
Locale(String language, String country, String variant)
```

2.2.2 Unicode

Um dos requisitos fundamentais da internacionalização de uma aplicação é, naturalmente, o suporte à introdução e visualização de caracteres que não apenas os do alfabeto latino [Int02]. O *standard* Unicode representa o estado da arte no que respeita à codificação universal de caracteres, abrangendo todos os sistemas de escrita do mundo [Con09]. O Unicode é implementado por um UTF – *Unicode transformation format* – que mapeia cada um dos valores numéricos Unicode para uma sequência de *bytes* única. Existem três esquemas de codificação UTF que fazem parte do *standard* Unicode: UTF-8, UTF-16 e UTF-32.

2.3 Conteúdo localizado

Na perspectiva da internacionalização e da localização, o conteúdo de um produto é composto por diferentes tipos de componentes, que podem ser categorizados [Sun01]:

- **Funcionalidade** — Diz respeito ao domínio da aplicação. Os produtos de *software* são concebidos por forma a realizarem algum tipo de função. Os aspectos funcionais de um produto têm influência na sua estrutura, sendo desejável que a sua internacionalização seja planeada de início.
- **Mensagens** — Informação textual do produto para o utilizador, de natureza tipicamente estática. Podem subdividir-se em mensagens internas (ex.: de *debug*, ou quaisquer outras destinadas aos programadores, pessoal de suporte técnico, de vendas, etc.) e externas, que são visíveis a qualquer utilizador, como mensagens de erro e de ajuda. As primeiras não devem ser localizadas e, como tal, não devem ser internacionalizadas.
- **Elementos gráficos de interface** — Janelas, botões, ícones, gráficos. A internacionalização destes elementos deve, idealmente, fazer parte da fase de desenho da interface, uma vez que a sua localização é por norma mais complexa do que a de mensagens. Por exemplo, é preferível desenhar ícones que possam ser usados mundialmente com a mesma conotação do que desenhar um ícone novo para cada mercado.

Conhecer esta taxinomia e identificá-la nos produtos é importante na medida em que as categorias não têm todas a mesma prioridade de internacionalização e localização. Das categorias apresentadas, a funcionalidade tem a prioridade mais alta, seguida das mensagens e, por fim, dos elementos gráficos de interface, que têm a prioridade mais baixa.

2.4 Processo de localização

Em [Ray04a], é analisado o fluxo de trabalho de um projecto de localização nas suas várias fases. Relativamente ao fluxo tradicional, é proposta uma metodologia alternativa no sentido de substituir os vários formatos de documentos relativos a um produto de *software* enviados para tradução (ficheiros *resource*, ficheiros de ajuda em HTML, manuais de utilização em edição electrónica, etc.) por um único formato. As diferentes fases deste processo na perspectiva da empresa de *software* estão sintetizadas na Figura 2.2.

A primeira fase, *Document creation*, é a criação dos documentos que irão necessitar de tradução. Neste contexto, documento deve ser entendido num sentido lato. A segunda fase, *Translation preparation*, consiste em extrair o texto traduzível dos documentos originais e convertê-los para um formato comum. Este formato é por isso considerado

Figura 2.2: Fluxo de trabalho de um processo de localização, tal como proposto em [Ray04a]

um formato intermediário no processo. O autor advoga, em particular, uma linguagem XML para este formato - XML Localisation Interchange File Format (XLIFF). Feita esta preparação, a documentação é enviada nesse formato comum para tradução por uma ou mais agências de tradução ou tradutores independentes - *Translation provider*. Quando os documentos devolvidos já traduzidos, ocorre um processo designado por *Reverse conversion* e que consiste na introdução das *strings* traduzidas nos documentos originais.

A Figura 2.3 apresenta o processo de tradução com mais detalhe e a um nível mais baixo [Ray04b].

O passo 1. representa a extracção do texto traduzível do documento original, tal como referido anteriormente. Este processo separa aquilo que é conteúdo traduzível propriamente dito do que não é, através de uma ferramenta para o efeito. O conteúdo traduzível é armazenado, neste caso, num ficheiro XLIFF, enquanto que as restantes porções são armazenadas num outro tipo de formato especial. Designamos essas porções por **esqueleto**. Por exemplo, num documento HTML existe muita informação que não é traduzível (embora possa ser relevante para a tradução em si), nomeadamente toda a estrutura de anotação. Assim, enquanto que um par de *tags* $<\text{title}>$ e $</\text{title}>$ é esqueleto, o seu conteúdo é texto traduzível. A figura introduz ainda o conceito de memória de tradução (*translation memory*). Uma memória de tradução é uma implementação de reciclagem de conteúdo, isto é, reutilização de conteúdo já traduzido [Int02]. Funciona como uma base de dados que associa segmentos de texto-fonte a uma ou mais traduções em uma ou mais línguas-alvo [Ope08]. Na figura, a memória de tradução intervém: no passo 2. para enriquecimento do ficheiro XLIFF gerado no passo anterior com traduções já existentes às quais os tradutores humanos terão acesso; no passo 5., no qual é a memória de tradução que é enriquecida com novas traduções realizadas pelos tradutores humanos. A reciclagem de conteúdo é uma técnica importante para a obtenção de custos mais reduzidos e períodos de localização mais curtos [Int02].

2.4.1 XLIFF

XLIFF é um dos formatos mais relevantes usados na indústria da localização [Ray04a]. A versão 1.2 foi aprovada como *standard* OASIS em Fevereiro de 2008, mas a sua versão inicial remonta a 2001 [Ope08]. Num documento XLIFF, o conteúdo localizável está presente em elementos $<\text{trans}-\text{unit}>$ (*translation units*). Dentro de cada unidade de tradução, reside um elemento para o texto-fonte, $<\text{source}>$, e outro para o texto-alvo, $<\text{target}>$

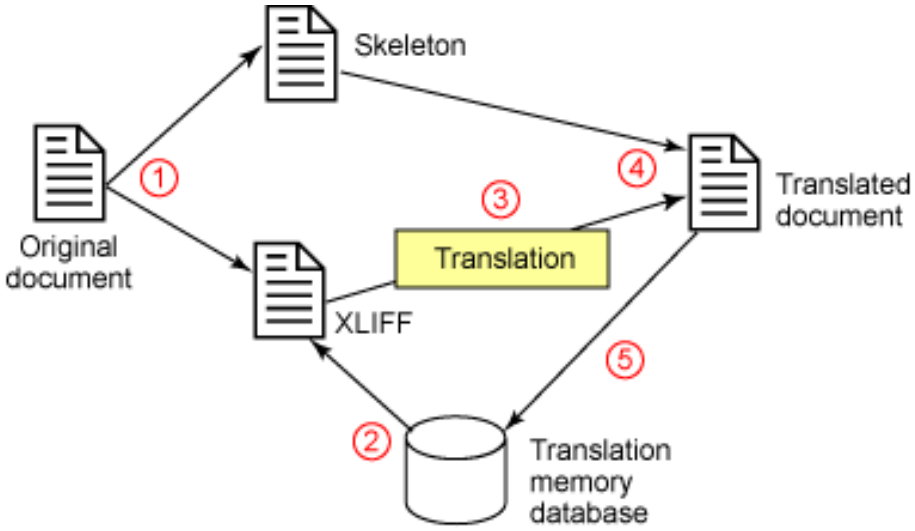

Figura 2.3: Processo de tradução.

[Ope08]. A listagem seguinte exemplifica o aspecto de uma tradução de inglês para francês de França em XLIFF:

```
<trans-unit id="1">
 <source xml:lang="en">Cannot find the file.</source>
 <target xml:lang="fr-FR">Fichier non trouvé.</target>
</trans-unit>
```

Capítulo 3

Descrição do Problema

3.1 Arquitectura do ALERT®

Na Figura 3.1, é apresentada a decomposição horizontal em camadas da arquitectura lógica do ALERT® [1].

Relativamente às decisões tecnológicas, temos:

- Base de dados: Oracle Database;
- Lógica de negócio: PL/SQL, predominantemente; alguma lógica é programada em Java;
- *Middleware* (Aplicação): Java, Apache Tomcat;
- Apresentação: Adobe Flash.

A persistência é assegurada por um sistema de gestão de base de dados Oracle. Para além de dados ditos "transaccionais", são também persistidos dados de "parametrização", uma vez que para além dos dados propriamente ditos se encontra outro tipo de informação que é usado na apresentação, como mensagens e descrições. A modelação é, essencialmente, relacional, embora sejam usadas algumas características objecto-relacionais implementadas pela Oracle. O acesso a dados e a implementação da lógica de negócio é feita em PL/SQL. A lógica de algumas funcionalidades, como a geração de relatórios, por exemplo, é feita em Java. A camada de *middleware* reside num servidor aplicacional Apache Tomcat e é responsável pela ligação das camadas que lhe estão adjacentes, bem como por aspectos transversais como segurança e gestão de conexões à base de dados. De uma forma resumida, a camada de apresentação invoca serviços disponibilizados pela camada

[1] Quando não especificado, entende-se por "ALERT®"o conjunto dos produtos clínicos, que partilham a mesma arquitectura e cujos lançamentos são feitos em simultâneo.

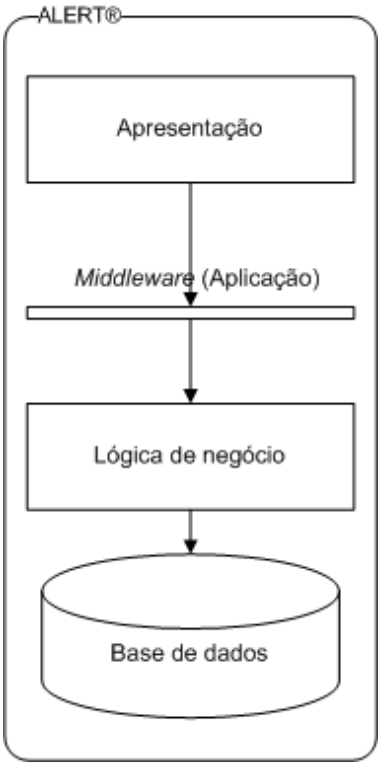

Figura 3.1: Decomposição horizontal da vista lógica da arquitectura do ALERT®.

aplicacional através de *Flash Remoting*. Estes serviços correspondem a invocações de métodos sobre objectos Java que, por sua vez, executam procedimentos/funções PL/SQL na base de dados.

As equipas de desenvolvimento organizam-se dividindo os recursos segundo cada uma das camadas/tecnologias, isto é, uma determinada equipa inclui programadores de base de dados e de interface com o utilizador e, eventualmente, de *middleware*.

3.2 Internacionalização do ALERT®

Na Figura 3.2, está representada a parte do modelo de dados relativa à internacionalização do ALERT®, versão 2.4.3[2].

É nestas tabelas que está contido na prática todo o conteúdo localizado do produto. Estas tabelas contêm as traduções das tabelas de parametrização, cujos dados correspondem na essência a configurações do produto [3]. Estes dados de parametrização distinguem-se pelo facto de fazerem parte do produto e portanto da sua instalação numa instituição, por oposição aos dados transaccionais.

[2] A última versão lançada, à data deste relatório.

[3] Em rigor, as tabelas que contêm as traduções são também elas próprias tabelas de parametrização.

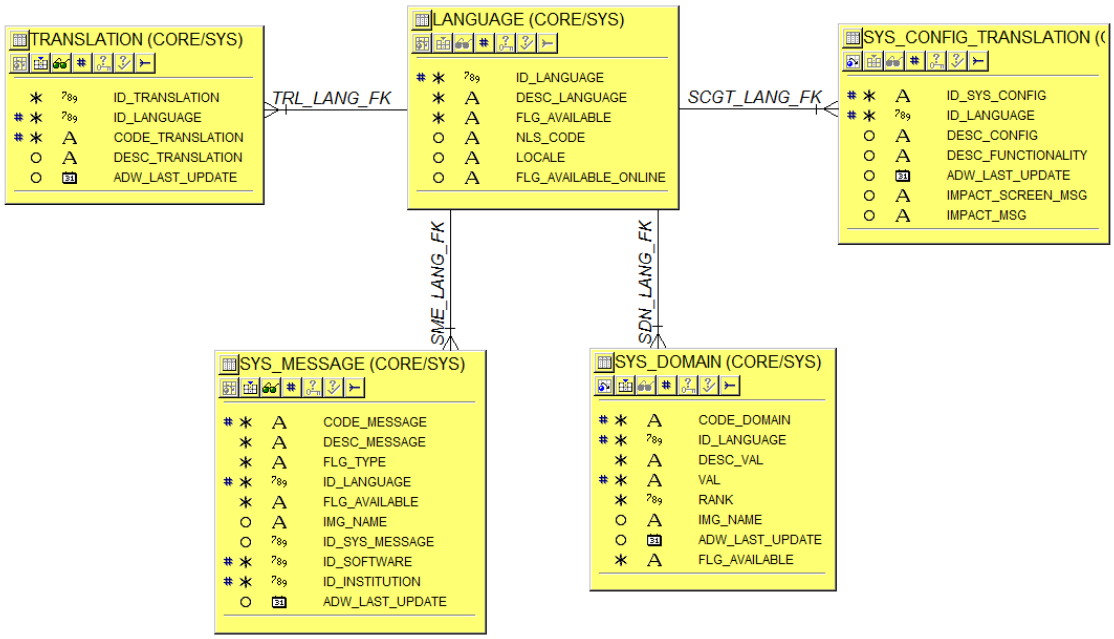

Figura 3.2: Internacionalização do modelo de dados do ALERT®. Colunas que compõem chave primária assinaladas com "#". Tipos de dados: "A" para tipos de caracteres e "789" para tipos numéricos.

Não surpreendentemente (cf. Cap. 2), os esquemas destas tabelas têm em comum um atributo para a língua (coluna ID_LANGUAGE), que faz sempre parte da chave primária, e um ou mais atributos correspondentes a descrições nessa língua (colunas de prefixo DESC_, ou também sufixo _MSG no caso da tabela SYS_CONFIG_TRANSLATION).

Relativamente à tabela TRANSLATION, esta contém, de uma forma geral, as traduções de todas as parametrizações do produto nas várias línguas. Uma vez que estas parametrizações estão em diferentes tabelas, que representam as diferentes entidades, ao ter-se pretendido agrupar a maioria das traduções nesta tabela não foram usadas restrições de integridade referencial. Antes, é usado um esquema de códigos para se fazerem referências. Confronte-se esta decisão de desenho com aquela que foi apresentada no Cap. 2, onde cada tabela teria associada uma tabela de detalhes, relativa à informação dependente de *locale*.

ID_COUNTRY	CODE_COUNTRY	CODE_NATIONALITY	ALPHA2_CODE	TAX
840	COUNTRY.CODE_COUNTRY.840	COUNTRY.CODE_NATIONALITY.840	US	11,000000
620	COUNTRY.CODE_COUNTRY.620	COUNTRY.CODE_NATIONALITY.620	PT	0,210000

Figura 3.3: Uma instância da tabela COUNTRY na qual estão representados os países E.U.A. e Portugal.

ID_LANGUAGE	CODE_TRANSLATION	DESC_TRANSLATION
1	COUNTRY.CODE_COUNTRY.620	Portugal
2	COUNTRY.CODE_COUNTRY.620	Portugal
1	COUNTRY.CODE_COUNTRY.840	Estados Unidos Da América
2	COUNTRY.CODE_COUNTRY.840	United States of America
1	COUNTRY.CODE_NATIONALITY.620	português
2	COUNTRY.CODE_NATIONALITY.620	Portuguese
1	COUNTRY.CODE_NATIONALITY.840	estado-unidense
2	COUNTRY.CODE_NATIONALITY.840	American

Figura 3.4: As traduções, em português de Portugal e inglês, correspondentes às entradas da Figura 3.3, na tabela TRANSLATION (não são mostradas todas as colunas).

Para exemplificar, regressemos ao ALERT® e tomemos o exemplo da tabela COUNTRY, que armazena a informação relativa à entidade país, tal como apresentado na Figura 3.3. Na modelação de dados do ALERT®, esta entidade tem dois atributos que constituem conteúdo traduzível: o nome do país (coluna CODE_COUNTRY) e a designação da nacionalidade de um indivíduo desse país (coluna CODE_NATIONALITY). Essas colunas, ditas de código, com nome de prefixo CODE_, seguem o seguinte esquema de codificação [4]:

<nome da tabela>.<nome da coluna>.<identificador do registo> [ALE08b]

Como se pode ver, tem um formato reflexivo sobre a própria tabela, coluna e linha. É através destes códigos que se faz a referenciação entre a tabela de parametrização e a tabela de traduções (neste caso COUNTRY e TRANSLATION, respectivamente). Na tabela TRANSLATION, a coluna que contém o código, CODE_TRANSLATION, forma chave primária juntamente com a língua. Podemos inferir que, provavelmente, a base de dados sofreu um processo de internacionalização posterior ao seu desenho inicial e que estas colunas que actualmente contêm códigos, continham anteriormente o próprio texto, numa só língua.

Algumas categorias de conteúdo localizado presentes na tabela TRANSLATION são:

- Conteúdo clínico. Ex.: designações de alergias, análises, diagnósticos, exames, intervenções, etc;
- Mecanismos de interface. Ex.: *templates*, que são um tipo de formulário próprio do produto;
- Texto presente em determinados elementos da interface, como botões;
- De uma forma geral, todo o conteúdo que necessita de tradução e não se encontra nas restantes tabelas.

[4] Relativamente a um esquema de codificação, deveria haver alguma *validação*, através de restrições do tipo *check* na DDL SQL, por exemplo, ainda que seja possível fazer a sua *verificação* junto da realidade do modelo [Cel05].

CODE_MESSAGE	DESC_MESSAGE	FLG_TYPE	ID_LANGUAGE	FLG_AVAILABLE	IMG_NAME	ID_SOFTWARE	ID_INSTITUTION
ALLERGY_LIST_T005	Alergias |Categoria 1	A	1	Y		0	0
ALLERGY_LIST_T005	|Categoria 1	A	3	Y		0	0
ANALYSIS_M001	Já existem análises com resultado associada a esta requisição.	E	1	Y		0	0
ANALYSIS_M001	Ya existen análisis con resultado asociado a esta solicitud.	E	3	Y		0	0
AllergyCreateStep01Selection	Este ecrã apresenta as alergias organizadas por categorias. Para	H	1	Y		0	0
AllergyCreateStep01Selection	Ecrã de registo de alergias. Para registar uma alergia , seleccion	H	1	Y		12	0
AllergyCreateStep01Selection	Esta pantalla presenta las alergias organizadas por categorías. Pa	H	3	Y		0	0
PREV_EPISODE_T150	Relatório detalhado de episódio de Consulta	R	1	Y		0	0
PREV_EPISODE_T150	Informe detallado de la consulta	R	3	Y		0	0

Figura 3.5: Uma instância da tabela SYS_MESSAGE, em português de Portugal e espanhol (não são mostradas todas as colunas).

Relativamente à tabela SYS_MESSAGE, "contém diferentes tipos de texto a mostrar nas aplicações ALERT®, como títulos de campos e mensagens para o utilizador" [ALE08b]. Ao contrário da tabela TRANSLATION, esta tabela contém apenas traduções das próprias entidades que modela — mensagens de sistema. Ao colocarmos os atributos dependentes de *locale* na mesma tabela que modela a mensagem, está-se em a considerar que existe uma relação um-para-um entre a entidade mensagem e esses atributos, em vez de uma relação um-para-muitos (cf. Cap. 2). Dito de outra forma, todas as colunas de SYS_MESSAGE estão a depender de ID_LANGUAGE, ainda que isso não pareça corresponder à realidade. Nesta tabela, CODE_MESSAGE tem apenas um papel de identificador da mensagem, enquanto que DESC_MESSAGE mantém o significado de texto da mensagem na respectiva língua. Relativamente às outras colunas:

- FLG_TYPE: H - ecrãs de ajuda, A - títulos e headers aplicacionais, E - mensagens de erro, R - mensagens incluídas nos relatórios;
- FLG_AVAILABLE: disponibilidade do registo ('Y'es, 'N'o);
- ID_SOFTWARE: aplicação ALERT® para a qual é válido o registo (0 significa qualquer aplicação);
- ID_INSTITUTION: instituição para a qual é válido o registo (0 significa qualquer instituição);
- IMG_NAME: nome da imagem arquivada nas livrarias do Flash, para os casos em que o título de uma coluna não é representado em texto, mas através de uma imagem representativa.

[ALE08b]

A tabela SYS_DOMAIN define domínios de valores relativos a determinados conceitos e as suas descrições em cada uma das línguas. Ao nível da aplicação, esses domínios de valores servem normalmente para popular listas de escolha múltipla. Por exemplo, relativamente a requisições de consultas, estas podem ter um de diversos estados em determinado momento. Este estado é indicado na coluna FLG_STATUS da respectiva tabela, CONSULT_REQ, cuja descrição, por sua vez, reside na tabela SYS_DOMAIN. A referenciação entre tabelas é feita através da mesma forma que aquela que se mostrou relativamente à tabela TRANSLATION. Neste caso a chave primária é composta, para além

da língua, por CODE_DOMAIN e por VAL. O valor de CODE_DOMAIN é preenchido com:

<nome da tabela>.<nome da coluna da tabela> [ALE08b]

relativamente à tabela referenciada. A Figura 3.6 exemplifica, usando o domínio dos estados de requisições de consultas.

Relativamente à tabela SYS_CONFIG_TRANSLATION, esta diz respeito a descrições de configurações de sistema das aplicações ALERT® para diferentes instituições, que por sua vez estão definidas na tabela SYS_CONFIG.

Com efeito, "Na tabela SYS_CONFIG_TRANSLATION é inserido um registo por cada ID_SYS_CONFIG [da tabela SYS_CONFIG] e para cada idioma com os descritivos da configuração e da funcionalidade em que é aplicada" [ALE08b]. Apesar disso, mais uma vez não é garantida integridade referencial através de uma restrição do tipo chave estrangeira. Segundo o modelo de dados, uma configuração tem quatro atributos passíveis de tradução, que correspondem às colunas DESC_CONFIG, DESC_FUNCTIONALITY, IMPACT_SCREEN_MSG e IMPACT_MSG. A Figura 3.7 exemplifica com uma configuração.

Tendo em consideração a taxinomia apresentada no Cap. 2 respeitante ao conteúdo de um produto de *software* na perspectiva da internacionalização e localização, e perante o que foi apresentado anteriormente, pode-se estabelecer a seguinte classificação:

- **TRANSLATION**: funcionalidade.
- **SYS_MESSAGE**: funcionalidade, mensagens (externas), elementos gráficos de interface.
- **SYS_DOMAIN**: funcionalidade, elementos gráficos de interface.
- **SYS_CONFIG_TRANSLATION**: mensagens (internas).

Os elementos de interface dizem respeito às colunas IMG_NAME, que definem o nome de uma imagem a carregar. Este trabalho não incide sobre a localização deste tipo de componentes.

Relativamente às mensagens internas presentes em SYS_CONFIG_TRANSLATION, estas estão internacionalizadas e são localizadas, como vimos, apesar de [Sun01] recomendar o contrário, como também já havíamos visto no Cap. 2.

No que toca ao peso relativo que cada tabela representa no universo de toda a informação traduzível presente no produto, podemos estimar [5] que a tabela TRANSLATION representa o maior peso, seguida das tabelas SYS_MESSAGE, SYS_DOMAIN e SYS_CONFIG_TRANSLATION, por esta ordem.

[5] Com base no número de linhas de cada tabela, em ambiente de desenvolvimento. Para uma avaliação mais fidedigna, dever-se-ia considerar a palavra como unidade de peso, bem como deveria ser usado um ambiente mais estável para efectuar esse levantamento (cf. Sec. 3.3.3).

ID_CONSULT_REQ	FLG_STATUS
1048	M
2325	R
2336	A
1068	F
1069	F
1070	C
2342	R
1075	C

CODE_DOMAIN	ID_LANGUAGE	DESC_VAL	VAL
CONSULT_REQ.FLG_STATUS	1	Agendada	M
CONSULT_REQ.FLG_STATUS	1	Requisitado	R
CONSULT_REQ.FLG_STATUS	1	Pedido lido	F
CONSULT_REQ.FLG_STATUS	1	Por agendar	P
CONSULT_REQ.FLG_STATUS	1	Resposta lida	A
CONSULT_REQ.FLG_STATUS	1	Cancelado	C
CONSULT_REQ.FLG_STATUS	1	Autorizado	T
CONSULT_REQ.FLG_STATUS	1	Aprovado	V
CONSULT_REQ.FLG_STATUS	1	Processado	S
CONSULT_REQ.FLG_STATUS	1	Pedido Rejeitado	N
CONSULT_REQ.FLG_STATUS	6	Autorisé	T
CONSULT_REQ.FLG_STATUS	6	Demandé	R
CONSULT_REQ.FLG_STATUS	6	Non programmé	P
CONSULT_REQ.FLG_STATUS	6	Programmé	M
CONSULT_REQ.FLG_STATUS	6	Réponse d'interprétation	A
CONSULT_REQ.FLG_STATUS	6	Annulé	C
CONSULT_REQ.FLG_STATUS	6	Demande refusée	N
CONSULT_REQ.FLG_STATUS	6	Approuvé	V
CONSULT_REQ.FLG_STATUS	6	En cours d'exécution	S
CONSULT_REQ.FLG_STATUS	6	Demande d'interprétation	F

Figura 3.6: Uma instância da tabela CONSULT_REQ, acima, mostrando requisições de consultas em diferentes estados. Abaixo, as descrições dos diversos valores do domínio em português de Portugal e francês, na tabela SYS_DOMAIN.

ID_SYS_CONFIG	VALUE	DESC_SYS_CONFIG
ALERT_INTF_TIMEOUT	24	Tempo em horas definido para deixar de mostrar os alertas de Interfaces

ID_SYS_CONFIG	ID_LANGUAGE	DESC_CONFIG	DESC_FUNCTIONALITY	IMPACT_MSG	IMPACT_SCREEN_MSG
ALERT_INTF_TIMEOUT	1	Tempo em horas definido para deixar de mostrar os alertas de Int	Alertas		
ALERT_INTF_TIMEOUT	2	Time (in hours) to stop displaying interface alerts.	Alerts		
ALERT_INTF_TIMEOUT	3	Tiempo definido en horas para dejar de mostrar las alertas de Inte	Alertas		
ALERT_INTF_TIMEOUT	5	Tempo predefinito in ore per terminare la presentazione degli avvi	Avvisi		
ALERT_INTF_TIMEOUT	8	Cas (v hodinách) sa prestal zobrazovat v alert prepojeniach	Alerts: Upozornenia		
ALERT_INTF_TIMEOUT	11	Tempo em horas definido para deixar de mostrar os alertas de Int	Alertas		

Figura 3.7: Acima, a definição de uma configuração na tabela SYS_CONFIG. Abaixo, o conteúdo localizado respectivo, na tabela SYS_CONFIG_TRANSLATION, em várias línguas.

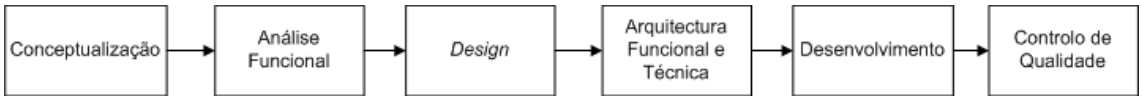

Figura 3.8: Fases de desenvolvimento na ALERT.

3.3 Processo de Desenvolvimento

3.3.1 Metodologia

Na Figura 3.8 apresentam-se de uma forma geral as fases de desenvolvimento de *software* presentes na metodologia da ALERT. Pode-se dizer que se aproxima de um modelo em "cascata" [6] , na medida em que o desenvolvimento é faseado e o progresso para uma fase seguinte está dependente da conclusão e aprovação dos resultados da fase actual [Wei03]. Por exemplo, em princípio os programadores apenas começam a desenvolver tendo desenhos aprovados (fase de *Design*) bem como documentação de arquitectura.

3.3.2 Versões

O desenvolvimento é orientado à versão e, no âmbito da versão, à funcionalidade. Uma nova versão do ALERT® é planeada por forma a que um conjunto de novas funcionalidades passem a fazer parte do produto no fim do ciclo de desenvolvimento relativo a essa versão.

O ciclo de desenvolvimento de uma versão não acaba com o fecho da versão. Para resolver *bugs* que são detectados depois do fecho da versão ou, de uma maneira geral, efectuar desenvolvimentos sobre uma versão fechada são criados *fixes* para essa versão. Desta forma, é de notar que podem existir, em simultâneo, pelo menos duas linhas de desenvolvimento: versão "actual" e *fixes* da versão anterior, por exemplo. Caso o próprio *fix* necessite de correcções, pode ainda existir um *fix* sobre esse *fix*, nesse caso normalmente designado por *hotfix*.

Relativamente à identificação das versões, é usado um esquema de sequências de números inteiros separados por pontos. Assim:

- Uma versão é identificada por uma sequência de 3 ou menos números. Ex.: 2.4.3 e 2.5.
- A versão 2.5 é posterior à versão 2.4.3, por exemplo.
- Números mais à esquerda na sequência representam maior importância ao nível das mudanças no produto.

[6] A existência de tal modelo e/ou a sua definição é disputada. Veja-se, por exemplo, [Wei03].

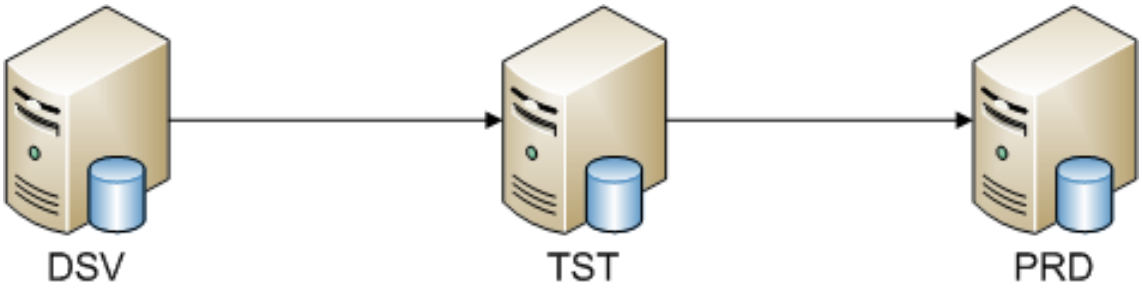

Figura 3.9: Os ambientes que suportam a metodologia usada na ALERT: de desenvolvimento (DSV), de testes (TST) e de produção (PRD).

- Um *fix* é identificado acrescentando um número à sequência da versão respectiva de forma incremental. Ex.: 2.4.3.5 representa o quinto *fix* sobre a versão 2.4.3.

3.3.3 Ambientes de Desenvolvimento

Existem três ambientes que suportam o desenvolvimento de *software*, tal como mostra a Figura 3.9. Um ambiente não é mais do que uma instalação de todos os componentes do ALERT® cujo nível de maturação pode variar consoante o seu propósito.

O ambiente de desenvolvimento (DSV) é aquele sobre o qual a equipa de desenvolvimento trabalha diariamente, desenvolvendo novas funcionalidades ou efectuando correcções. Neste ambiente, não é garantida a estabilidade do código ou a qualidade dos dados.

No ambiente de testes (TST) trabalha a equipa de controlo de qualidade (sendo por isso também designado por essa forma), validando as funcionalidades e dados. Neste ambiente deverão já existir parametrizações de determinados universos como, por exemplo, conteúdo clínico. O código é considerado estável — a equipa de desenvolvimento não pode efectuar alterações, sendo da responsabilidade da equipa de versionamento a execução de código nesse ambiente.

O último ambiente, ambiente de produção (PRD), contém a última versão certificada do produto e é a partir deste ambiente que são realizadas as implementações nos clientes.

Existe ainda um ambiente usado para demonstrações, que normalmente se baseia em cópias do ambiente de produção.

Tendo em conta as linhas de desenvolvimento por versão, são também mantidos estes ambientes por versão.

3.3.4 Processo de Localização

A criação de conteúdo localizado na aplicação e a sua tradução desenrola-se, tipicamente, da seguinte forma:

- A equipa de desenvolvimento de base de dados recebe desenhos aprovados e documentos de arquitectura, que são artefactos de saída das fase de *Design* e de Arquitectura Funcional e Técnica, respectivamente. Os primeiros são protótipos não-funcionais da interface com o utilizador e funcionam como *storyboards* de uma determinada funcionalidade a desenvolver.

- Com base nessas especificações, pode ser necessário introduzir novo conteúdo na base de dados. Essa introdução consiste, como vimos na Sec. 3.2, em inserções nas tabelas que modelam as respectivas entidades e nas tabelas que armazenam as traduções.

- Quando o desenvolvimento de uma funcionalidade é considerado terminado e estável, é necessário versionar o respectivo código, incluindo as inserções de conteúdo localizado associadas. Estas inserções são versionadas através de *scripts* que consistem em chamadas a procedimentos PL/SQL que executam, essencialmente, MERGEs SQL nas tabelas de traduções. Os ficheiros versionados têm o seguinte aspecto (um exemplo para cada tabela):

```
insert_into_translation(1,'DOC_ELEMENT_CRIT.CODE_ELEMENT_CLOSE.4598'
    ,'Respiração - apneia');

insert_into_sys_message(1,'V_ALERT_M021','Pedido de consulta da
    especialidade @1 respondido','A',0,0);

insert_into_sys_domain(2,'EXAM_REQ_DET.FLG_STATUS','In progress',
    'E');

insert_into_syscfg_translation(i_lang =>1 ,i_sys_config =>
    'TO_FMT_YYYYMMDDHH24MISS',i_desc_config => 'ID SYS_CONFIG
    utilizado para obter o formato da data e hora', i_desc_func =>
    'Templates', i_impact_msg => 'Altera o formato de visualização
    das datas (data/hora) que sao efectuadas utilizando o formato
    de registo de dados do tipo Toque',i_impact_screen_msg => NULL);
```

- A equipa de desenvolvimento inclui este código num local específico do repositório SVN e requer o seu versionamento à equipa responsável, que o executa em TST. As execuções sem problemas são registadas com vista à construção de *scripts* finais de versão.

- A equipa de desenvolvimento elabora um ficheiro Microsoft® Excel com o texto a ser traduzido, normalmente a partir de uma exportação da base de dados. Este ficheiro contém algumas *macros* com o objectivo de facilitar posteriormente o processo de importação das traduções efectuadas.

- O documento é enviado para o departamento de línguas da empresa, por correio electrónico, e é requerida a sua tradução. No departamento de línguas, é seguido um processo interno com vista à produção do documento traduzido. A tradução é efectuada directamente sobre o documento Excel ou, eventualmente, usando o *software* de tradução assistida SDL Trados [SDL08]. Importa categorizar os tradutores da seguinte forma:
  - Internos à ALERT
    - Colaboradores do departamento de línguas (sede do grupo de empresas)
    - Colaboradores de subsidiária/distribuidora
  - Externos à ALERT
- O documento traduzido é devolvido à equipa de desenvolvimento, que importa o texto traduzido para a base de dados, caso seja relevante ter essas traduções em DSV, ou requer apenas o seu versionamento.

3.3.5 Limitações Identificadas

Relativamente ao processo de localização e à sua integração com o restante processo de desenvolvimento, a própria empresa identificou alguns problemas e limitações. Fazemos aqui uma resenha tendo em conta duas fontes: a equipa da área da arquitectura e [Eir08].

A equipa de arquitectura aponta os seguintes aspectos:

- Não existe uma instalação ALERT® (ambiente) com todo o conteúdo localizado;
- A sincronização dos diversos ambientes relativamente ao conteúdo localizado não é fiável;
- Não existe a garantia de que não sejam perdidas traduções no decorrer do processo;
- Muitas das vezes, o mesmo conteúdo — ou conteúdo idêntico — é traduzido mais do que uma vez;
- É necessário ter uma visão global e centralizada do conteúdo localizado.

Citando [Eir08], por seu turno:

- "A equipa de desenvolvimento elabora o modelo de dados e insere conteúdos nas novas tabelas. Muito desse conteúdo deve ter traduzido, embora outro conteúdo, como o clínico, ou a descrição dos edifícios das instituições não deve ser traduzido e versionado. Este tem que ser identificado;
- Não é possível saber em que versão do Alert® surgiu um texto.

- Durante a tradução, a equipa de traduções não tem contexto nem informação suficiente para realizar traduções adequadas e de qualidade;
- Por vezes, aparecem erros ortográficos e sintácticos;
- As traduções são geridas com ficheiros de Excel que exigem bastante mão-de-obra e os cujos dados podem ficar deprecados;
- Por vezes, uma listagem de textos para traduções sobrepõe uma listagem de textos anterior já traduzida previamente;
- Muitas vezes, as listagens para tradução de textos numa língua A não traduzidos numa língua B, inclui textos semelhantes que já foram previamente traduzidos nessa língua B;
- Muitas vezes, a tradução não é revista por consultores;
- A participação de indivíduos nas empresas afiliadas na tradução, é feita recorrendo a ficheiros Excel, e a tradução de uma listagem relativamente grande pode demorar meses, durante os quais esse ficheiro fica desactualizado. Por exemplo: a tradução completa do Alert® para uma nova língua implica a tradução de centenas de milhares de textos;
- Não há um repositório para todas as traduções efectuadas. Estas estão divididas entre o ambiente de desenvolvimento e a máquina de demonstrações comerciais e formação interna;
- Não há controlo de versões sobre cada texto traduzido. Não é possível obter de forma fácil um histórico de edições de um único texto."

Capítulo 4

Solução Proposta

A solução proposta consiste numa aplicação Web que tenha subjacente um repositório de conteúdos localizados independente do ALERT®, criando desta forma um ambiente centralizado no que diz respeito ao armazenamento dos dados e distribuído no que diz respeito ao processo localização dos produtos.

A apresentação da solução, neste capítulo, passa por uma visão geral, pela modelação do domínio do problema, análise de requisitos e modelação de dados.

4.1 Visão Geral

Nesta secção, olhamos para a arquitectura de alto nível da solução proposta.

A Figura 4.1 apresenta uma vista de distribuição. A aplicação para gestão de conteúdos localizados deve ser entendida como uma aplicação Web que reside numa máquina servidor da sede da empresa e que aloja os componentes de servidor Web, servidor aplicacional e servidor de base de dados. Os clientes da aplicação ligam-se através das suas máquinas dotadas de um navegador Web. Estes clientes são os colaboradores do departamento de desenvolvimento e tradutores do departamento de línguas, ligados directamente à rede da empresa, ou tradutores localizados fora da sede, que se ligam à rede através da internet por VPN, por exemplo.

A Figura 4.2 apresenta o fluxo da informação que passa pela aplicação no decorrer de um processo de localização. Do ambiente de desenvolvimento provém um ficheiro de conteúdo em língua original. A aplicação fornece mecanismos de importação desse conteúdo para o seu repositório. A aplicação expõe esse conteúdo para tradução, que pode ser efectuada directamente através da própria interface com o utilizador ou localmente, na máquina do tradutor. Neste caso, terá de haver mecanismos de importação/exportação

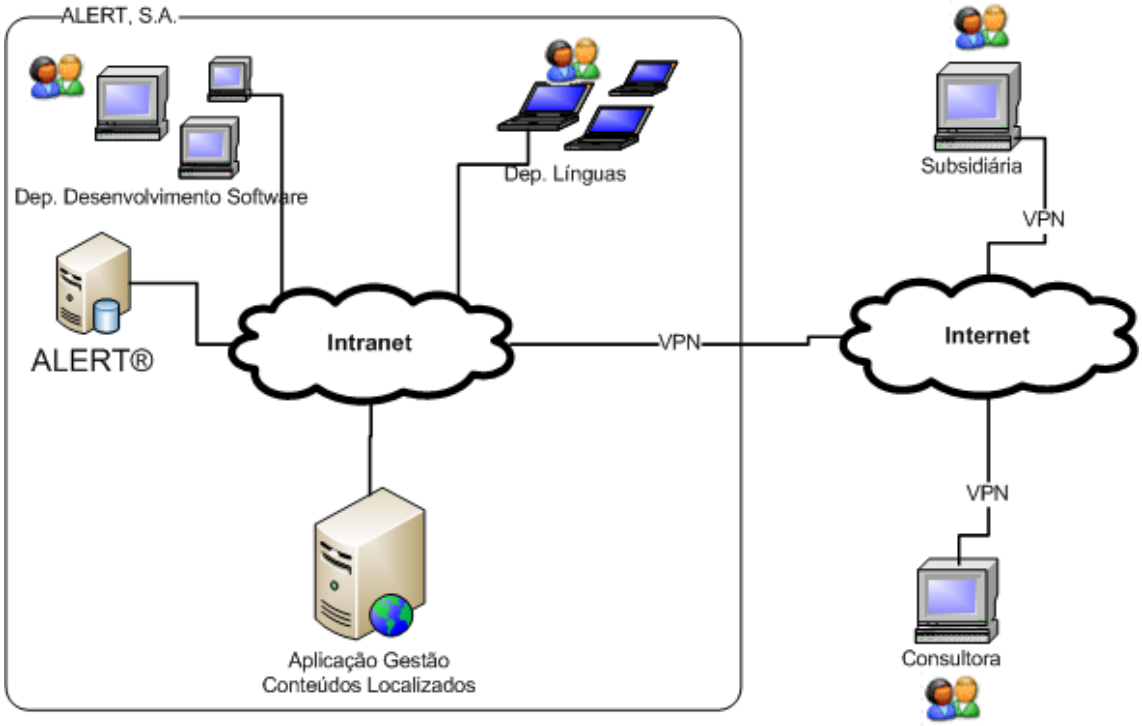

Figura 4.1: Arquitectura de alto nível da solução proposta.

intermédios para interoperação com as aplicações *desktop* usadas pelos tradutores. Finalmente, a aplicação fornece mecanismos de exportação do conteúdo traduzido, tendo em conta que este será instalado numa instância do ALERT®. A aplicação mantém todo o conteúdo localmente na forma de um modelo de dados relacional.

4.2 Modelação do Domínio e da Estrutura

Nesta secção apresenta-se, utilizando UML, a modelação do domínio do problema (do ponto de vista da análise de requisitos)[BS02, SV01] e da estrutura estática do sistema a desenvolver (do ponto de vista do desenho da solução)[SV01]. Inclui-se no Anexo A o diagrama de classes completo.

4.2.1 Entidades Básicas

Na Figura 4.3 estão representadas as entidades fundamentais do domínio.

A classe Item representa de forma abstracta uma determinada *string* que surge na interface de um produto ALERT®. Isto é, representa conteúdo localizado sem considerar nenhuma descrição textual em particular. Essa concretização em texto é modelada pela classe Tradução e daí a associação um-para-muitos entre Item e Tradução. Uma tradução é, naturalmente, uma tradução numa determina língua. Optando por esta modelação, o

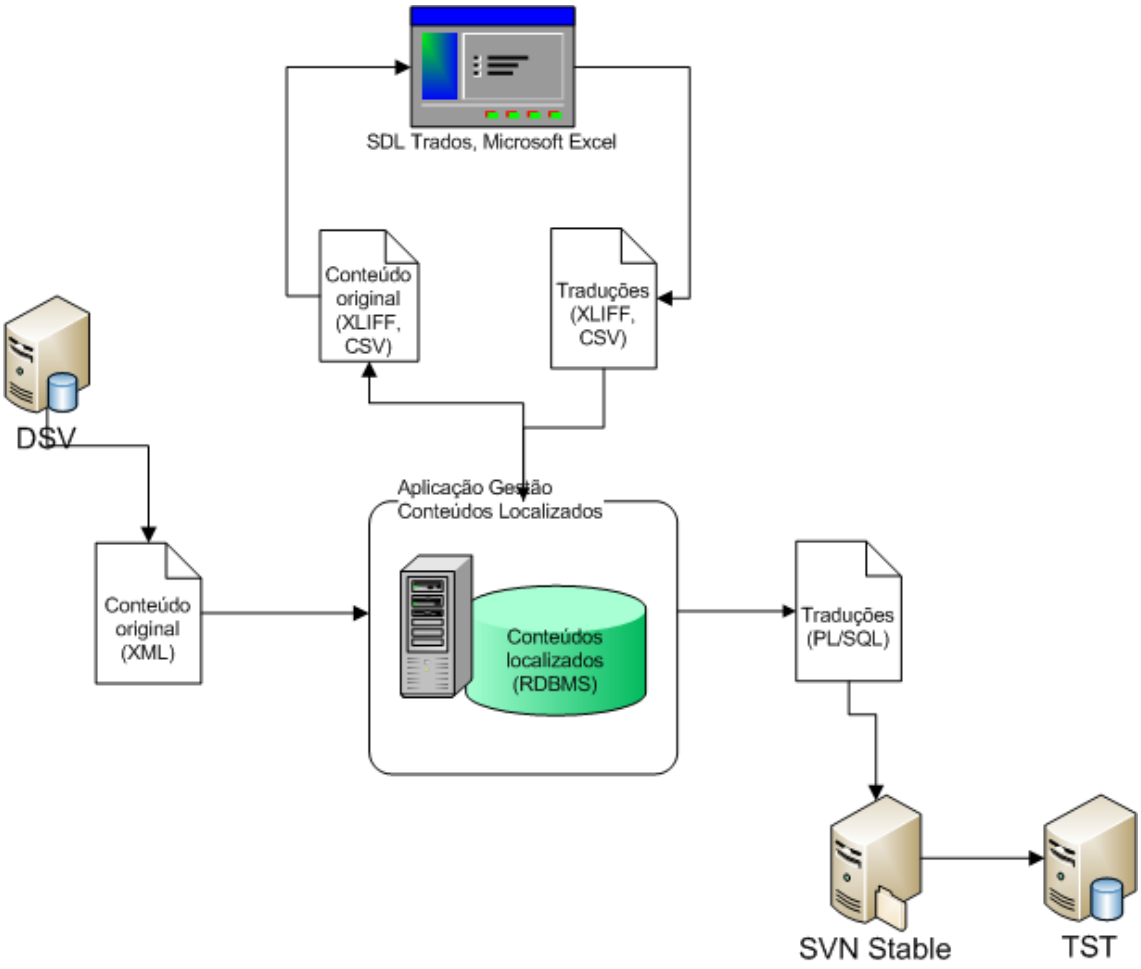

Figura 4.2: Fluxo de informação num processo de localização apoiado por uma aplicação dotada de um repositório centralizado.

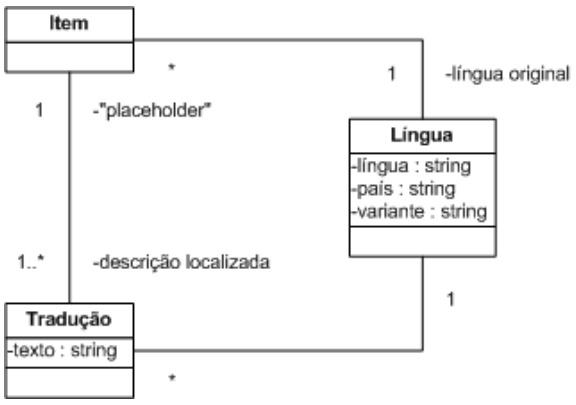

Figura 4.3: Diagrama de classes: entidades básicas.

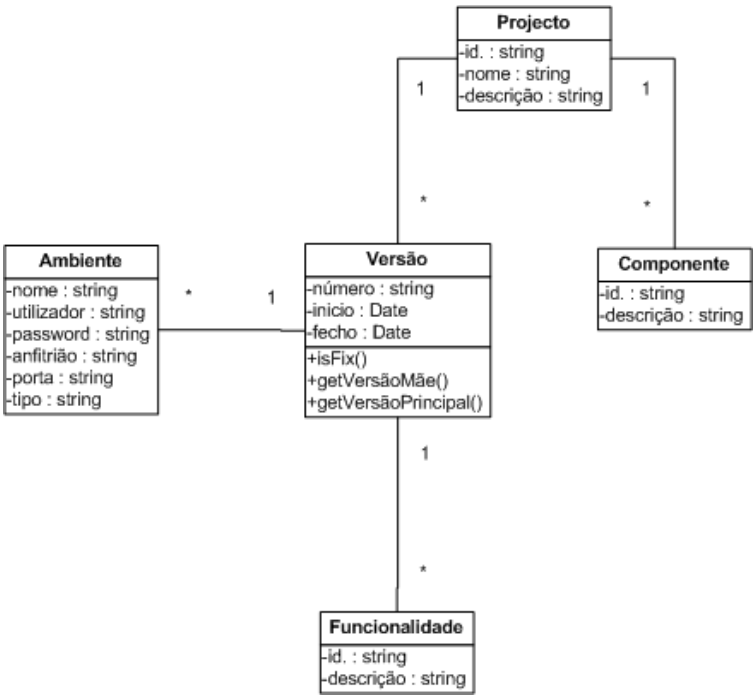

Figura 4.4: Diagrama de classes: processo de desenvolvimento.

texto original de um item é, portanto, idêntico a uma outra qualquer instância de Tradução. Conseguimos identificar essa tradução "especial" através da associação entre Item e Língua, que representa a língua original de um item. No contexto da produção de conteúdos localizados na ALERT®, essa língua será tipicamente aquela na qual são produzidos os desenhos da interface (cf. Sec. 3.3) e que constitui língua-fonte no processo de tradução. Considera-se que um item tem de ter pelo menos uma tradução e que, a ser única, essa tem de ser a tradução na língua original do item — o fundamento é que não fará sentido no âmbito de um processo de localização não existir nada tangível a traduzir. Tomando novamente como exemplo os países no modelo de dados do ALERT® (cf. Sec. 3.2), consideramos que o conceito de "nacionalidade de um indivíduo proveniente dos Estados Unidos Da América" é um item e que "American" e "estado-unidense" são os textos das suas traduções em inglês e português, respectivamente. Um item tem, assim, aproximadamente o mesmo significado que *translation unit* tem em XLIFF (cf. Cap. 2) ou que uma *entrada* tem num ficheiro .PO de GNU *gettext* [Fou08]. No nosso domínio, consideramos três atributos para definir a entidade Língua: língua, país e variante (cf. Cap. 2).

4.2.2 Processo de Desenvolvimento

Na Figura 4.4 estão representadas as entidades relacionadas com o processo de desenvolvimento.

A classe Projecto representa uma unidade de desenvolvimento dentro da empresa e, por isso, representa um determinado produto (ou conjunto de produtos se intimamente relacionados, como é o caso dos produtos clínicos). Por exemplo, o ALERT® — que como vimos nos capítulos anteriores pode ser entendido como um produto único ou como o conjunto dos produtos clínicos — é representado pelo projecto de nome "CLINICAL - ALERT Development" e identificador "ALERT". O produto ALERT® Personal Health Record (PHR), por exemplo, enquadra-se numa linhagem de produtos diferente e segue a sua própria linha de desenvolvimento. Logo, o projecto que lhe está associado é diferente — "CITIZEN - PHR". Podemos encontrar a entidade Projecto no *software* usado para gestão de tarefas e *bugs*, por exemplo, o que não só nos dá segurança relativamente à correcta modelação da realidade da empresa, como nos retira o trabalho de definir que instâncias da classe Projecto existem. A presença do conceito de projecto na estrutura do sistema é importante na medida em que permite suportar, sem adaptação, conteúdos localizados de outros produtos que sigam o esquema de internacionalização do ALERT®.

Um projecto pode ser decomposto nas seguintes subunidades: componentes e versões, e estas ainda em funcionalidades. Um componente pode ter um dos seguintes significados, que são entendidos internamente na empresa:

- Um produto do projecto. Relativamente ao projecto ALERT, poderíamos ter os componentes EDIS, ORIS, PRIVATE PRACTICE, ...
- Um módulo funcional do projecto, provavelmente transversal a vários produtos. Relativamente ao projecto do ALERT, poderíamos ter os componentes BACKOFFICE, CORE (cerne da aplicação), IMAGING (módulo de imagiologia), SCHEDULER (agenda), ...

Versão representa qualquer dos conceitos de versão, *fix* ou *hotfix* apresentados na Sec. 3.3.2. Apesar de haver uma hierarquia entre estes diferentes conceitos, considera-se que pertencem todos ao mesmo conjunto de entidades e, como tal, modelam-se com a mesma classe. No entanto, como continua a ser importante distinguir versões — ditas agora versões principais — de *fixes*, consideramos que a classe define operações com essa finalidade. A uma versão de um projecto está associado um conjunto de funcionalidades, como foi referido na Sec. 3.3.2. Por fim, a cada versão está ainda associado um conjunto de ambientes de diferentes tipos, como vimos na Sec. 3.3.3. A classe Ambiente tem os atributos necessários ao estabelecimento de uma ligação entre o sistema e a instância de base de dados que corre no ambiente.

É apresentada na Figura 4.5 a forma como se associam as classes das figuras 4.3 e 4.4.

Para caracterização dos itens, estes associam-se a um número arbitrário de funcionalidades e componentes. A associação com Funcionalidade, em particular, deve ser entendida da seguinte forma: um item foi criado no modelo de dados do ALERT® para suprir as necessidades de uma ou mais funcionalidades. Isto permite, por exemplo, orientar a

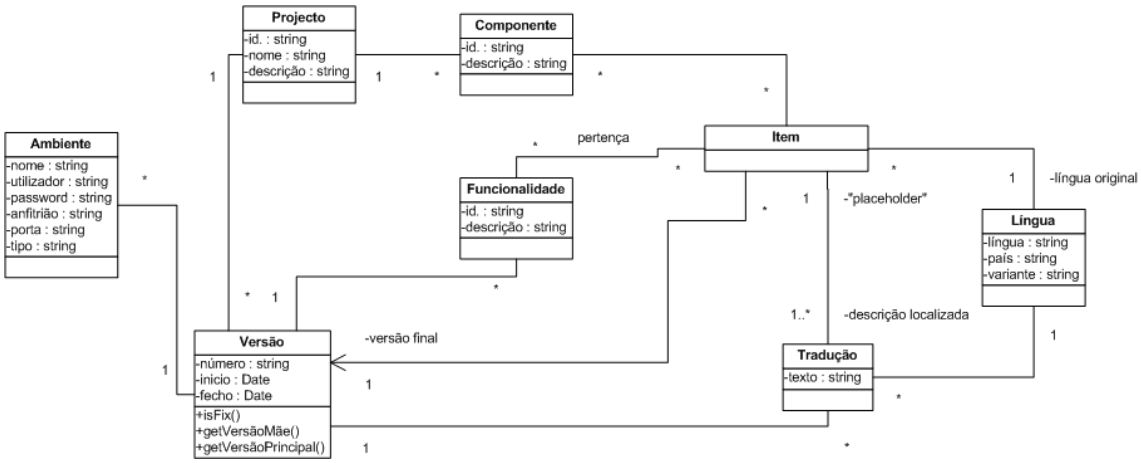

Figura 4.5: Diagrama de classes: associação entre as classes das figuras 4.3 e 4.4.

tradução à funcionalidade, aproximando os processos de desenvolvimento e localização. Com esta informação, é ainda possível saber em que versão do ALERT® surgiu um texto. Relativamente à associação com Componente, esta pode revelar-se mais importante para traduções em massa. Por exemplo, numa abordagem a um novo mercado é necessário traduzir para uma nova língua a totalidade do conteúdo de pelo menos um dos produtos, mas nem sempre para todos. Nestes casos, filtrar o universo dos itens por componente pode ser útil.

Finalmente, Item associa-se a Versão para representar a última versão na qual um item é utilizado, por forma a modelar o caso em que um conjunto de itens seja descontinuado, continuando a existir em versões mais antigas.

A associação muitos-para-um entre Tradução e Versão está relacionada com o ciclo de desenvolvimento na qual a tradução foi efectuada e relativamente ao qual vai ser versionada. Assim, para um mesmo item, será normal que as traduções nas várias línguas se distribuam por versões consecutivas, uma vez que dificilmente é possível realizar todas as traduções no mesmo ciclo. Para além disso, poderemos querer manter descrições diferentes do mesmo item na mesma língua, entre versões diferentes. A versão de uma tradução identifica, por isso, essa tradução.

4.2.3 Gestão de Traduções

Na Figura 4.6 introduzem-se os utilizadores que desempenham os papéis dos actores definidos na Sec. 4.3 e a forma como se associam ao conteúdo localizado.

Assim, consideramos que, do ponto de vista do sistema, um item pode ser criado e modificado por um determinado gestor de conteúdo, enquanto que a autoria das traduções pode ser da responsabilidade tanto de um tradutor como de um gestor de conteúdo, uma vez que a tradução de um item na língua original é da responsabilidade deste último.

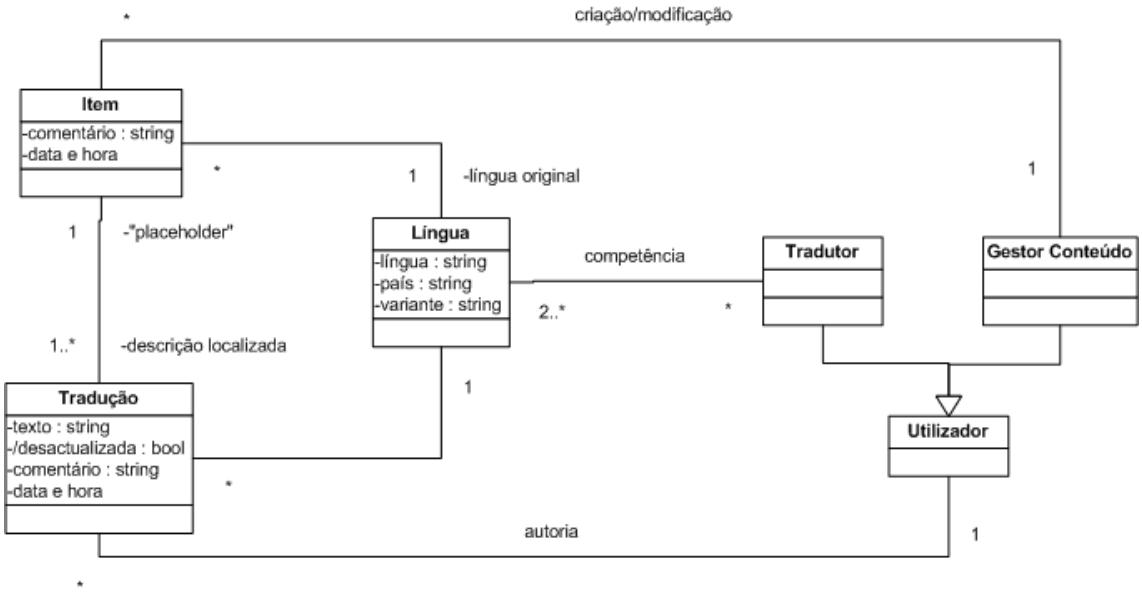

Figura 4.6: Diagrama de classes: responsabilidades dos utilizadores relativamente ao conteúdo localizado.

Acrescenta-se ainda a noção da competência linguística do tradutor em duas ou mais línguas. Às classes Item e Tradução acrescentam-se atributos relativos à data e hora da alteração, bem como um possível comentário por parte do utilizador responsável. Relativamente a uma tradução, poderemos ainda determinar se se encontra desactualizada, isto é, se a sua data é anterior à data da tradução da língua original do mesmo item.

Na Figura 4.7 é introduzido o conceito de pedido de tradução. Um gestor de conteúdo pode requerer a tradução de um determinado conjunto de itens, para uma ou mais línguas e a ter efeito numa ou mais versões (cf. Sec. 4.2.2).

4.2.4 Dicionário de Dados

Relativamente à forma de tratar o conteúdo localizado, optou-se, como se viu na Sec. 4.2.1, por uma modelação que se abstrai da forma como esse conteúdo está modelado e implementado no ALERT®. Isto faz sentido uma vez que no domínio do problema identificamos informação que pretendemos tratar da mesma forma — descrições textuais de determinados elementos em diferentes línguas — independentemente da forma como essa informação é persistida no ALERT®. No entanto, tal como nos processos de localização baseados em XLIFF é necessário o processo de extracção, bem como o esqueleto e a sua manipulação com vista à reconversão (cf. Cap. 2), também na estrutura do nosso sistema necessitamos de considerar os dados adicionais e os metadados associados ao conteúdo original com vista à importação e à exportação.

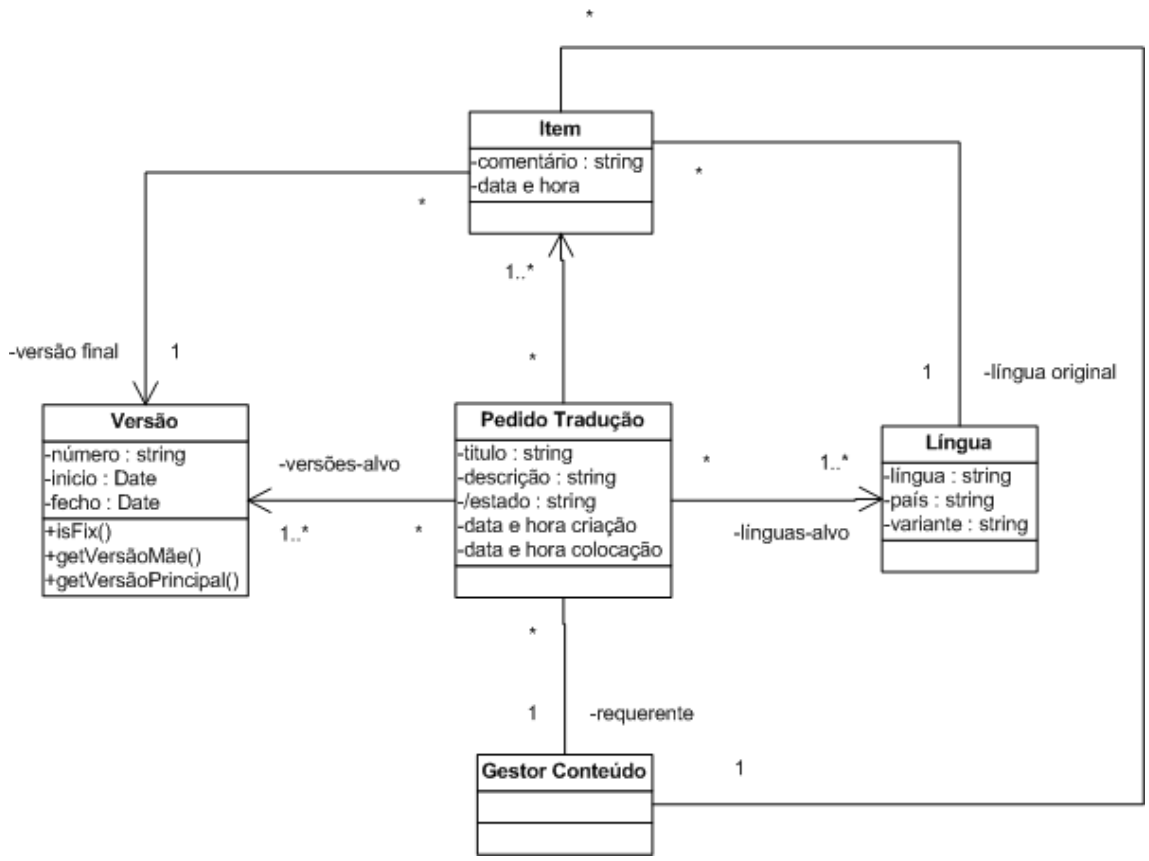

Figura 4.7: Diagrama de classes: gestão de traduções.

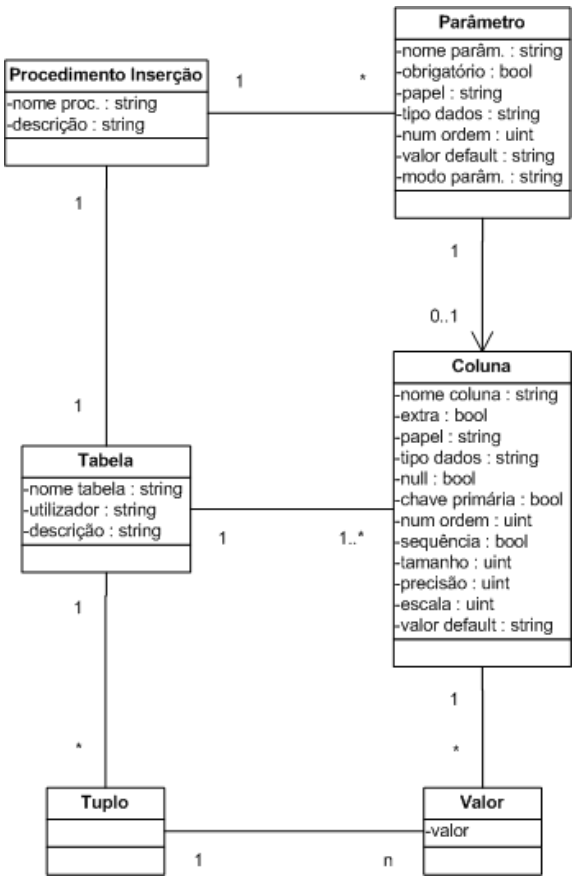

Figura 4.8: Diagrama de classes: dicionário de dados.

Na Figura 4.8 apresenta-se o conjunto de classes que pretende representar um dicionário de dados do conteúdo localizado em língua original tal como presente no modelo relacional do ALERT®.

As classes Tabela e Coluna e a associação entre elas representa o mesmo tipo de informação que poderemos encontrar numa real implementação de um dicionário de dados de um sistema de gestão de base de dados relacional. Relativamente aos atributos a manter, foi usado o dicionário de dados Oracle como modelo [Ora09]. As instâncias destas classes são as tabelas de traduções do ALERT® e respectivas colunas (cf. Cap. 3).

As classes Tuplo e Valor correspondem a uma modelação orientada a objectos das restantes entidades que encontramos no modelo relacional. Isto é, uma tabela contém um número arbitário de tuplos e cada tuplo tem um determinado número de valores, tantos quantas colunas a tabela tiver. Cada um desses valores é o valor de uma das colunas da tabela. As instâncias destas classes são as linhas das tabelas de traduções que têm conteúdo em língua original e os respectivos valores.

Esta estrutura seria já suficiente para servir de interface com qualquer base de dados do ALERT®. No entanto, como vimos no Cap. 3, são usados, por norma, procedimentos PL/SQL pré-definidos para versionamento e instalação de conteúdo localizado, em lugar de se utilizar directamente instruções SQL **INSERT INTO**. Assim, para podermos ter *output* concordante com esta prática, necessitamos ainda de considerar que para cada tabela está definido um procedimento de inserção PL/SQL. Este procedimento declara um determinado número de parâmetros que à partida correspondem aos dados a inserir em cada uma das colunas da tabela respectiva, no caso em que um parâmetro se associa a uma coluna. Deixa-se em aberto a possibilidade de pedirem outro tipo de informação, no caso em que um parâmetro não se associa a nenhuma coluna.

Dois atributos da classe Coluna em particular mantêm informação intrínseca ao nosso domínio: extra e papel. O primeiro diz-nos se uma determinada coluna pertence efectivamente a uma tabela de tradução do ALERT® ou se não pertence, sendo usada para veicular informação de entrada adicional. Por exemplo, relativamente ao conteúdo clínico, este é normalmente identificado por um código de uma norma internacional. Este código pode ser relevante para o processo por motivos vários, apesar de não constar de nenhuma tabela de tradução. Assim, é interessante poder requerer, ao nível da importação do conteúdo localizado, um formato mais alargado. Quanto ao atributo papel, dá-nos informação relativa ao significado da coluna da tabela de tradução no âmbito do nosso domínio, ou seja, relativamente ao item. Explicitando: na tabela TRANSLATION, por exemplo, pode verificar-se que a coluna ID_LANGUAGE tem o papel de identificador da língua da tradução do item, que DESC_TRANSLATION tem o papel de tradução do item nessa língua e que CODE_TRANSLATION tem o papel de identificador (chave) do item. Na tabela SYS_DOMAIN, ID_LANGUAGE tem o papel de língua, DESC_VAL tem o papel de tradução e CODE_DOMAIN e VAL têm o papel de chave. Podemos aplicar o mesmo

exercício às tabelas restantes. Estes papéis estabelecem relação entre o dicionário de dados e as entidades básicas do domínio (cf. Sec. 4.2.1). Podemos considerar que as demais colunas têm papel de esqueleto.

Resumindo, a estrutura relativa ao dicionário de dados serve de base à implementação da interoperabilidade do sistema com uma instalação ALERT®[1]. Em analogia com XLIFF, o dicionário de dados constitui o esqueleto do conteúdo localizado.

Na Figura 4.9 apresenta-se a associação do dicionário de dados ao restante modelo. Verifica-se que o modelo de dados do ALERT® que suporta as traduções pode sofrer alterações de uma versão para a outra, pelo que associamos Versão a Tabela — uma ou mais tabelas de traduções para cada versão de projecto. Desta associação retiramos ainda que uma tabela é identificada pela versão. Da descrição das tabelas de traduções do ALERT® no Cap. 3 verificamos que, no caso das tabelas TRANSLATION, SYS_MESSAGE e SYS_DOMAIN apenas uma coluna contém conteúdo localizado, enquanto que no caso de SYS_CONFIG_TRANSLATION, quatro colunas contêm conteúdo localizado. Assim, no caso geral, e como se pode ver pela Figura 4.10, um item corresponde, no modelo de dados do ALERT®, ao cruzamento de uma coluna com as linhas que partilham a mesma chave primária com excepção da coluna de língua. Isto é o mesmo que dizer que existe uma associação entre Item e Tuplo. A associação Item-Tuplo é um-para-muitos uma vez que, como foi referido, as tabelas são identificadas pela versão, mas um item é transversal às versões do projecto (embora seja criado numa versão e possa ser descontinuado a partir de outra).

Para completar a definição de uma estrutura que suporte mecanismos de importação e exportação, é ainda necessário considerar o mapeamento entre a identificação da língua no ALERT®, que é um número inteiro, e a identificação de língua do sistema. Considerando ainda que cada projecto pode ter um identificador de língua diferente, essa mapeamento está patente na classe-associação da Figura 4.11.

4.3 Análise de Requisitos

Foi efectuado um levantamento de requisitos do sistema com base no modelo para especificações de requisitos de *software* utilizando casos de utilização do Rational Unified Process [oT02], tendo sido produzido um documento [ALE09]. Assim, o resultado desse levantamento consistiu num modelo de casos de utilização [SV01, BS02]) contendo os principais casos de utilização para captar requisitos funcionais do sistema; e numa

[1] Ou com outro projecto, desde que persista também os seus conteúdos localizados numa base de dados relacional.

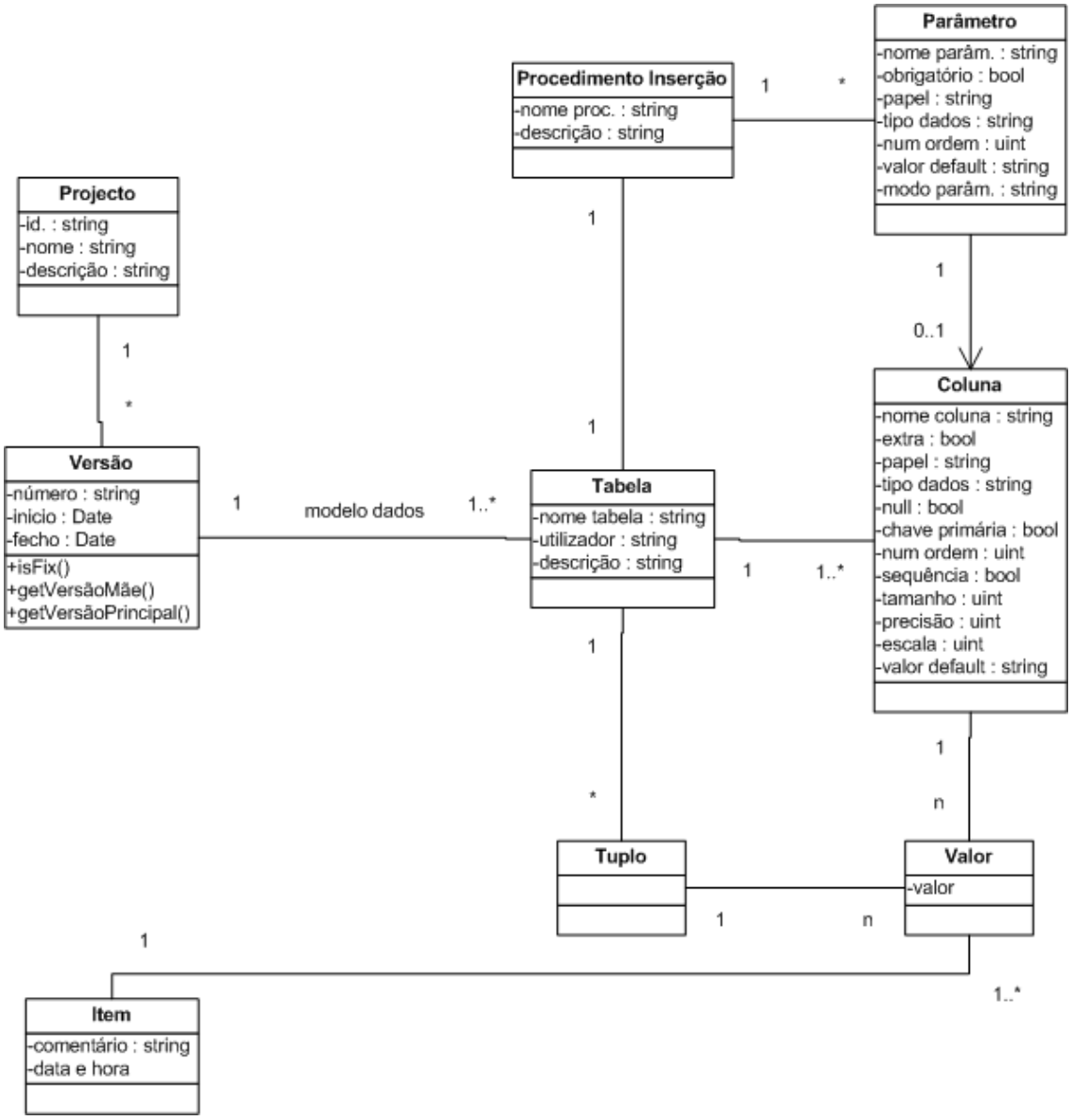

Figura 4.9: Diagrama de classes: associação entre o dicionário de dados e o restante modelo.

ID_SYS_CONFIG	ID_LANGUAGE	DESC_CONFIG	DESC_FUNCTIONALITY	IMPACT_MSG
ADMIN_ADMISSION	1	Alta de Urgência para Internamento nos EU	Alta	A alta de Urgência para internamei
ADMIN_ADMISSION	2	Admission discharge requires an administra	Discharge	Admission discharge will require th
APPLICATION_TIMEOUT	1	Timeout geral da aplicação (minutos)	Geral	Ao fim dos minutos indicados, a ap
DIGITAL_SIGNATURE	1	A assinatura digital está disponível?	Relatórios	Permite ou não a assinatura digital
DIGITAL_SIGNATURE	2	Is digital signature available?	Relatórios	Allows users to digitally sign docun
DIG_SIG_ALG_TYPE	1	Algoritmo usado na assinatura digital de dc	Relatórios	Altera o algoritmo usado para assi

Figura 4.10: Identificação da abstracção de item no modelo de dados do ALERT® (SYS_CONFG_TRANSLATION).

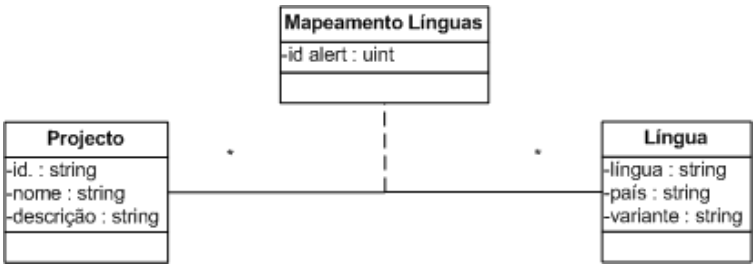

Figura 4.11: Diagrama de classes: mapeamento da identificação de línguas do sistema e de um projecto.

especificação de requisitos suplementares, contendo outros requisitos funcionais não captados pelo modelo de casos de utilização, ou transversais a este, bem como requisitos não-funcionais [oT02, BS02].

4.3.1 Modelo de Casos de Utilização

4.3.1.1 Vista Geral

Apresenta-se uma visão geral do modelo de casos de utilização.

A Figura 4.12 mostra os actores do modelo e as breves descrições dos actores são apresentadas nos parágrafos seguintes [BS02].

Actores

Gestor de Conteúdo O actor Gestor de Conteúdo é responsável pela gestão do conteúdo localizado. Introduz e mantém o conteúdo na sua língua original, requer a sua tradução para as restantes línguas e extrai as traduções do repositório.

Tradutor O actor Tradutor é responsável por realizar pedidos de tradução colocados pelo actor Gestor de Conteúdo e por manter traduções já realizadas, nas línguas para as quais tem competência.

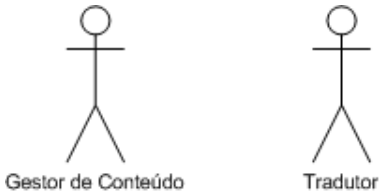

Figura 4.12: Actores do modelo.

4.3.1.2 Casos de Utilização

A Figura 4.13 mostra os principais casos de utilização do sistema. As breves descrições dos casos de utilização são apresentadas nos parágrafos seguintes [BS02]. Não são apresentadas as descrições completas dos fluxos de eventos dos casos de utilização.

Introduzir Conteúdo Localizado Este caso de utilização descreve como o actor Gestor de Conteúdo utiliza o sistema para inserir conteúdo localizado no repositório, isto é, um conjunto de itens e respectiva tradução na língua original. O conteúdo pode ser inserido através de importação de ficheiro.

Manter Conteúdo Localizado Este caso de utilização descreve como o actor Gestor de Conteúdo utiliza o sistema para manter a informação relativa ao conteúdo localizado. Isto inclui visualizar, modificar e eliminar itens presentes no repositório.

Colocar Pedido de Tradução Este caso de utilização descreve como o actor Gestor de Conteúdo utiliza o sistema para colocar um pedido de tradução por forma a que um actor Tradutor possa identificar directamente uma necessidade de tradução.

Realizar Pedido de Tradução Este caso de utilização descreve como o actor Tradutor realiza um pedido de tradução, traduzindo os itens que lhe estão associados. A tradução pode ser efectuada directamente na aplicação ou localmente.

Manter Traduções Este caso de utilização descreve como o actor Tradutor utiliza o sistema para traduzir o conteúdo localizado da língua original para línguas-fonte da sua competência. Este caso de utilização inclui também visualizar e modificar traduções de itens existentes no repositório.

Exportar Traduções Este caso de utilização descreve como o actor Gestor de Conteúdo exporta as traduções do conteúdo localizado a partir do repositório. Podem ser exportadas as traduções relativas a uma determinada versão, pedido de tradução ou conteúdo genérico, por língua e em diferentes formatos.

Navegar pelo Conteúdo Localizado Este caso de utilização descreve como os actores Gestor de Conteúdo e Tradutor navegam pelo conteúdo localizado para localizarem determinados itens. A navegação pode ser feita de várias formas: por componente ou por versão/funcionalidade; por pesquisa; por filtros pré-definidos ou criados pelo próprio utilizador.

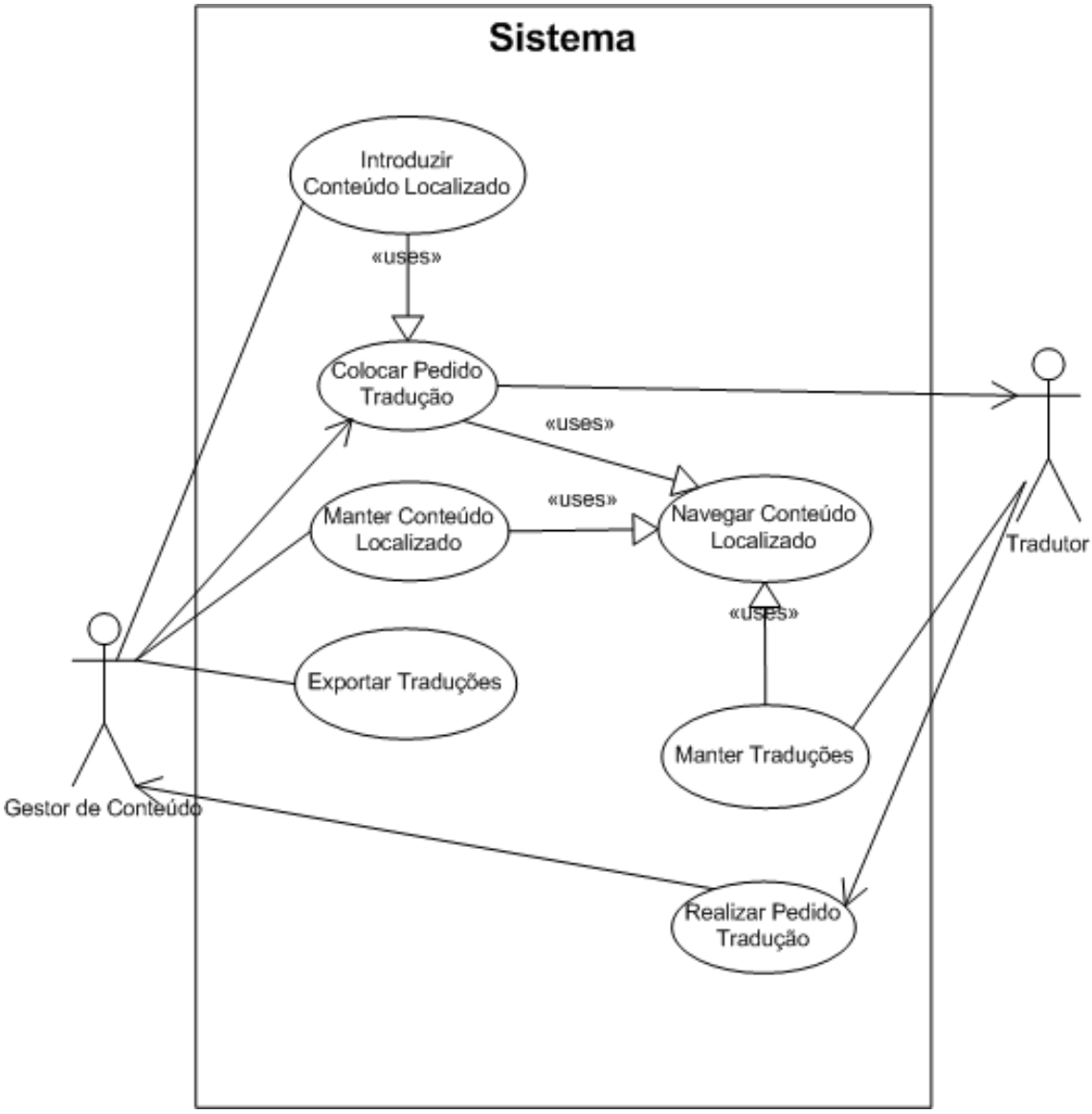

Figura 4.13: Diagrama de casos de utilização.

4.3.2 Requisitos Suplementares

4.3.2.1 Funcionais

Backoffice A aplicação deverá suportar um backoffice acessível a utilizadores com permissão de administração e que forneça as seguintes funcionalidades do tipo CRUD:

- Manutenção de utilizadores;
- Manutenção de ambientes;
- Manutenção do dicionário de dados;
- Manutenção de filtros.

Formatos de ficheiros suportados Deverão ser suportados os seguintes formatos de ficheiro:

- Importação
  - XML
  - CSV
- Tradução
  - Pelo menos um formato compatível com Microsoft® Excel
  - Pelo menos um formato compatível com SDL Trados 2007
- Exportação
  - Script PL/SQL
  - SQL*Loader
  - CSV
  - XML

Controlo de acessos A utilização da aplicação apenas deverá ser possível mediante autenticação perante a mesma. A aplicação deverá suportar um esquema de acessos baseado em perfis, que podem ser acumulados. Ver também Sec. 4.3.2.3.

Controlo de revisões Todas as manipulações ao nível de item ou de tradução deverão ser registadas, criando assim um histórico para controlo de múltiplas revisões. Entende-se por manipulação qualquer operação de inserção, alteração ou eliminação. Cada entrada do histórico deverá fornecer as seguintes informações:

- Utilizador responsável pela manipulação;
- Tipo de manipulação (inserção, alteração, eliminação);
- Data e hora da manipulação;
- Método usado (exs.: manual, importação, *rollback* do histórico);
- Os próprios dados.

Deverá ser possível, com base no histórico, reverter o repositório para um estado anterior, ao nível do item ou tradução.

Verificação ortográfica O texto do conteúdo localizado, ou seja, a tradução, deverá ser sujeito a verificação ortográfica quando a manipulação for manual (ou seja, introdução directa na aplicação). Se pelo menos uma tradução não passar na verificação ortográfica, o utilizador deverá ser alertado para esse facto e dever-lhe-á ser dada a oportunidade de prosseguir a operação ou de efectuar correcções.

4.3.2.2 Usabilidade

Internacionalização Tendo em conta o propósito da aplicação, esta deverá suportar a representação da maioria dos sistemas de escrita existentes, nomeadamente no que toca à correcta apresentação dos respectivos caracteres e à correcta orientação do texto. Relativamente à orientação do texto, deverá ser dada especial atenção às situações em que é necessário suportar texto bidireccional. Um exemplo desta situação será o caso em que se pretenda efectuar uma tradução da língua inglesa para a língua árabe.

4.3.2.3 Restrições de desenho

Arquitectura A aplicação deverá poder ser acedida por clientes utilizando um navegador Web.

O repositório da aplicação deverá ser implementado recorrendo a base de dados Oracle 10g.

Interface com JIRA A aplicação deverá integrar com o software Atlassian JIRA para manutenção de:

- Projectos;
- Componentes;
- Versões.

Interface com Active Directory A aplicação deverá integrar com Active Directory para realizar a autenticação de utilizadores.

4.4 Modelo de Dados

Usando o modelo UML apresentado na Sec. 4.2 como base para a conversão para o modelo relacional [UW02], obtemos um modelo de dados conforme apresentado na Figura 4.14[2]. Seguem-se alguns aspectos do modelo que merecem explicação.

Relativamente à tabela items, é usada uma chave de sistema como chave primária (id_item), mas ao mesmo tempo é mantida a sua chave natural. Esta chave consiste no conjunto das colunas name_column, encoded_key, name_table, owner_table e id_project. Aplica-se, portanto, uma restrição **UNIQUE** sobre este conjunto de colunas[3]. A coluna encoded_key, em particular, representa o mesmo elemento de dados que a coluna da tabela dbdd_rows com o mesmo nome que, como podemos ver, faz parte da chave primária dessa tabela. Este elemento consiste na reunião dos valores das colunas que têm o papel de chave do item (cf. Sec. 4.2.4), codificados de tal forma que constituam um valor atómico. A chave natural de items consiste, então, na reunião da informação relativa ao projecto, tabela, linha e coluna do modelo de dados do projecto que, como também vimos na Figura 4.10, identifica um item no universo dos conteúdos localizados da empresa. Por uma outra perspectiva, podemos verificar que a chave natural de items é igual à chave de dbdd_values com excepção da coluna relativa à versão, o que também é consistente com o que dissemos na Sec. 4.2.4. Manter correctamente a unicidade dos itens no modelo de dados da aplicação a desenvolver é importante, uma vez que um dos requisitos é a centralização da informação no repositório.

Na tabela languages é igualmente usada uma chave delegada como chave primária, uma vez que esta não pode conter colunas com **NULL** como valor, mas a sua chave natural é o conjunto das colunas language, country e variant, sendo as duas últimas opcionais, em concordância com o que vimos no Cap. 2.

O tipo de dados da coluna text da tabela translations, que contém o texto da tradução propriamente dito, necessita de ser declarado explicitamente como NVARCHAR, para armazenamento de *strings* de caracteres Unicode, caso a base de dados não seja criada em Unicode.

[2] Ao contrário do modelo do domínio, é usada a língua inglesa para as designações dos elementos de dados, não devendo no entanto ser difícil estabelecer a necessária correlação.

[3] As restrições de unicidade não são captadas pelo diagrama do modelo de dados.

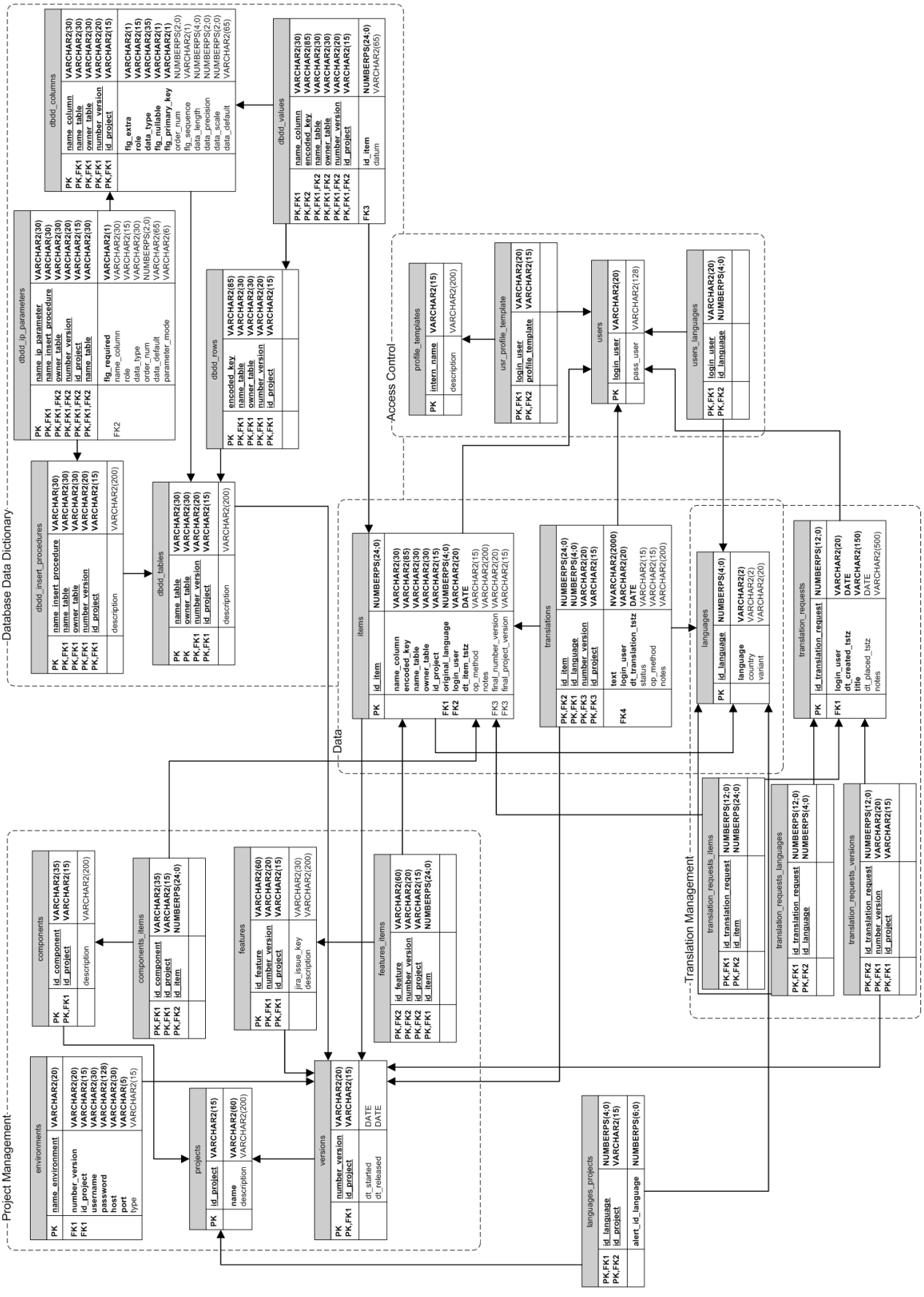

Figura 4.14: Diagrama do modelo de dados.

Capítulo 5

Conclusões e Trabalho Futuro

5.1 Satisfação dos Objectivos

Relativamente aos resultados esperados, foram cumpridos objectivos relativos à análise e especificação de uma solução, não tendo sido, contudo, realizada uma implementação.

Ainda assim, foi proposta uma aplicação com o objectivo de suportar o processo de tradução dos conteúdos localizados para as línguas dos mercados geográficos dos produtos ALERT®. O trabalho de especificação teve por base um estudo prévio do estado da arte no domínio do problema que lançou bases teóricas e tecnológicas relevantes, bem como uma análise do problema do ponto de vista técnico e processual no que toca à localização dos produtos clínicos ALERT®. A especificação do repositório ao nível do modelo de dados teve, assim, em conta a necessidade da substituição de dependências do modelo de dados do ALERT® e lança as bases para a implementação de uma entidade independente que interopere com esse mesmo modelo. Com efeito, o modelo de dados do ALERT® não foi afectado por este projecto. Conceitos intrínsecos ao processo de desenvolvimento foram também tidos em consideração, como forma de suportar, nomeadamente, a evolução do produto segundo diferentes linhas de desenvolvimento com o lançamento de novas versões e possíveis evoluções ao nível do modelo de dados do ALERT®. O repositório acrescenta ainda informação de categorização do conteúdo localizado que não existe no modelo de dados do ALERT®, por forma a providenciar maior contextualização e orientação no universo do conteúdo.

O projecto foi recebido na empresa como uma oportunidade de ver o seu *modus operandi* submetido a uma avaliação crítica que, por provir do exterior, seria naturalmente menos tendencial que uma visão interna. O trabalho efectuado foi tido em consideração e o projecto ganhou naturalmente o seu espaço, enquadrando-se com outros projectos em curso relacionados com a reestruturação da gestão da informação dos produtos.

5.2 Trabalho Futuro

No que concerne às perspectivas de trabalho futuro, é esperada uma implementação progressiva da especificação proposta e a integração da ferramenta daí resultante no trabalho diário dos diferentes colaboradores da empresa com intervenção no processo. O objectivo inicial de concretização de um sistema único utilizado para a tradução e gestão de conteúdos aplicacionais localizados mantém a sua importância, ao mesmo tempo que a empresa continua a investir na sua projecção internacional.

Foi também decidido que a implementação seria efectuada recorrendo à *framework* Web interna, relativamente recente e em franco desenvolvimento, potenciando uma decisão estratégica da empresa e promovendo a partilha e reutilização de conhecimento entre equipas de desenvolvimento diferentes.

Referências

[ALE08a] ALERT Life Sciences Computing, S.A. ALERT Life Sciences Computing — Company presentation / July 2008, July 2008.

[ALE08b] ALERT Life Sciences Computing, S.A. Database — code conventions and guidelines. DBCodeConventionsAndGuidelines_PROC_v1.1_PT.doc, 2008.

[ALE09] ALERT Life Sciences Computing, S.A. Aplicação para gestão de traduções — documento de análise de requisitos. AplicacaoGestaoTraducoes_v0.5x_PT.doc, 2009.

[Alv01] H. Alvestrand. Tags for the Identification of Languages. RFC 3066 (Best Current Practice), January 2001. Obsoleted by RFCs 4646, 4647.

[BS02] Kurt Bittner e Ian Spence. *Use Case Modeling*. Addison Wesley, 2002.

[Cel05] Joe Celko. *Joe Celko's SQL Programming Style*. Morgan Kaufmann, 2005.

[Con09] Unicode Consortium. The unicode consortium: Home page, 2009. http://www.unicode.org/.

[Eir08] João Carlos Martins Eiras. Simulador de casos clínicos do Alert® Paper Free Hospital. Master's thesis, Faculdade de Engenharia da Universidade do Porto, 2008.

[FEU08] FEUP. Proposta de projecto/dissertação: Gestão de conteúdos localizados dos produtos clínicos ALERT®, 2008.

[Fou08] Free Software Foundation. GNU 'gettext' utilities, 2008. http://www.gnu.org/software/gettext/manual/.

[Int02] Dr. International. *Developing International Software*. Microsoft Press, 2nd edition, 2002.

[Ope08] opentag.com, 2008. http://www.opentag.com.

[Ora09] Oracle. Oracle Database Online Documentation 10g Release 2 (10.2), 2009. http://www.oracle.com/pls/db102/homepage.

[oT02] Malmö University School of Technology. Rational unified process, 2002. http://www.ts.mah.se/RUP/RationalUnifiedProcess/.

[Ray04a] Rodolfo M. Raya. XML in localisation: A practical analysis. *IBM developerWorks*, Aug 2004.

[Ray04b] Rodolfo M. Raya. XML in localisation: Use XLIFF to translate documents. *IBM developerWorks*, Oct 2004.

[SDL08] SDL. Trados home, 2008. http://www.trados.com/.

[SSJtET02] Inderjeet Singh, Beth Stearns, Mark Johnson e the Enterprise Team. *Designing Enterprise Applications with the J2EE Platform*. Addison-Wesley, second edition, 2002.

[Sun01] Sun. Sun software product internationalization taxonomy. Technical report, Sun Microsystems, Inc., 2001.

[Sun08] Sun. Java Platform, Standard Edition 6 API Specification, 2008. http://java.sun.com/javase/6/docs/api/.

[SV01] A. Silva e C. Videira. *UML, Metodologias e Ferramentas CASE*. Centro Atlântico, Lda., 2001.

[UW02] Jeffery D. Ullman e Jennifer Widom. *A First Course In Database Systems*. Prentice Hall, 2002.

[Wei03] Conrad Weisert. There's no such thing as the Waterfall Approach! (and there never was). Information Disciplines, Inc., Chicago, February 2003.

Anexo A

Modelo do Domínio

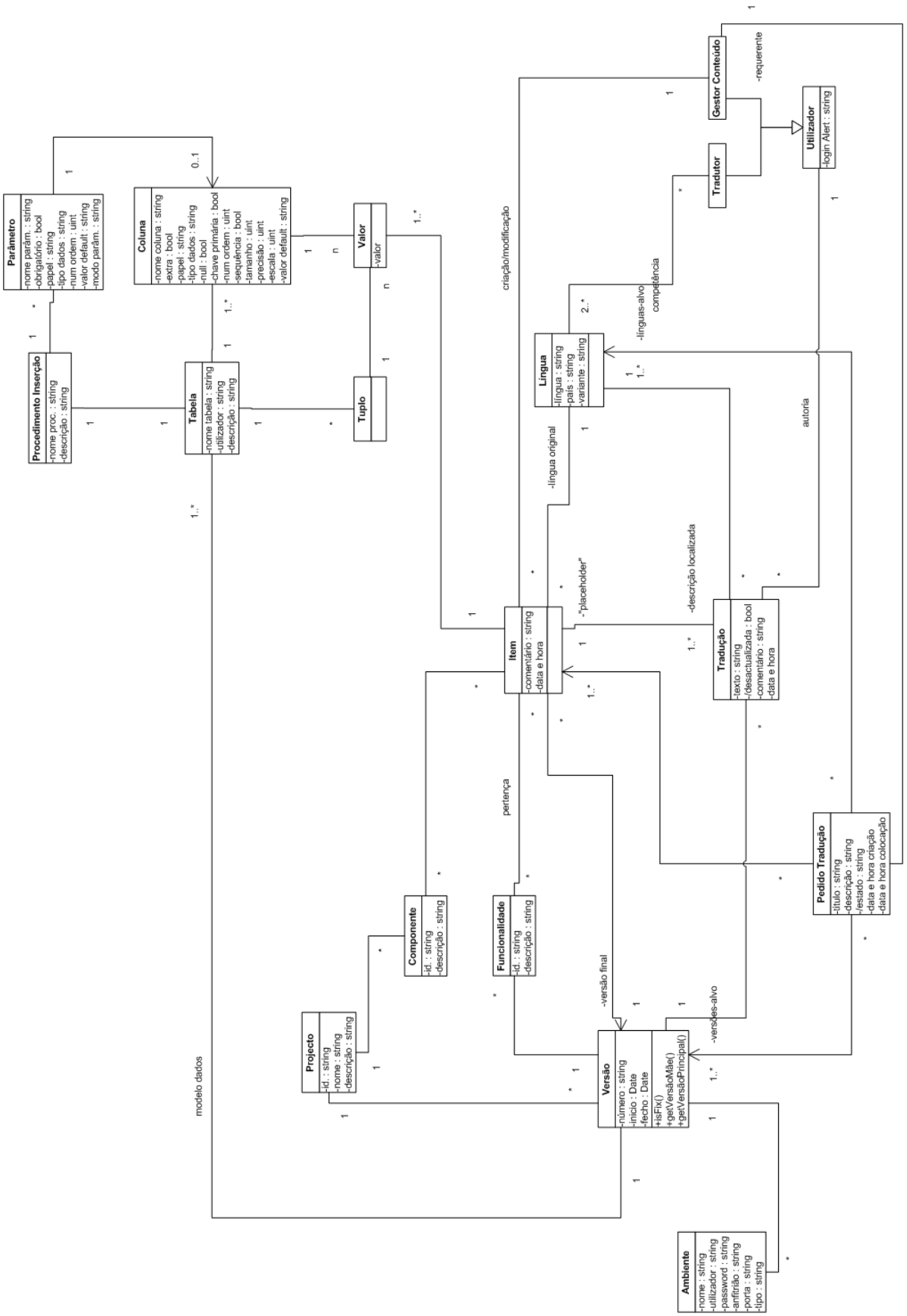

Figura A.1: Diagrama de classes do domínio.

Anexo B

Protótipos de Interface

Durante o projecto, foram elaborados alguns protótipos de interface com o utilizador em HTML e CSS por forma auxiliar o processo de análise de requisitos. Apresentam-se alguns deles ainda que representem apenas propostas de abordagem.

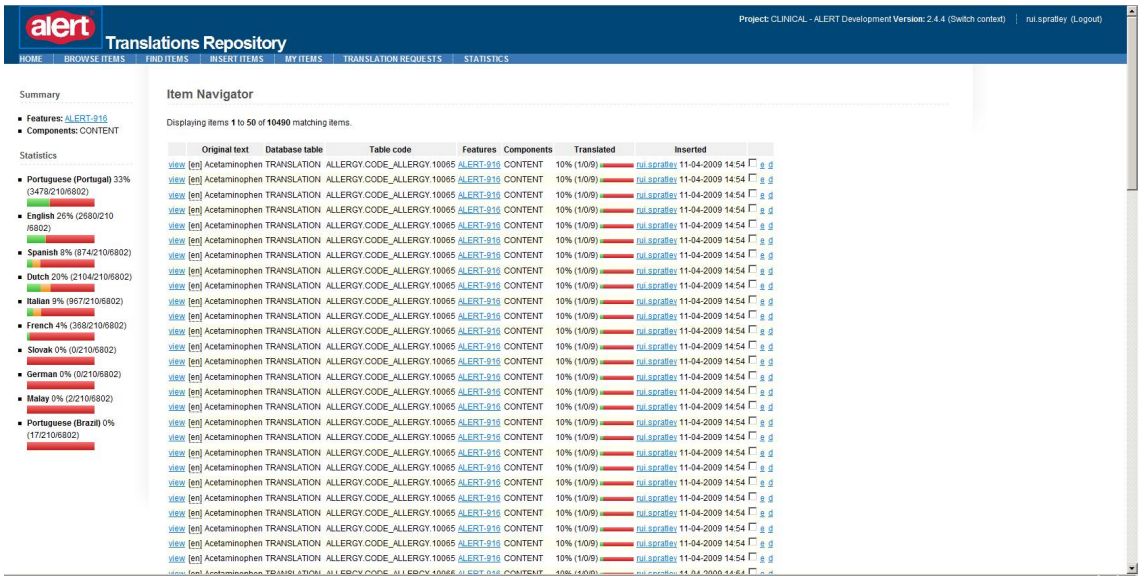

Figura B.1: Listagem de itens.

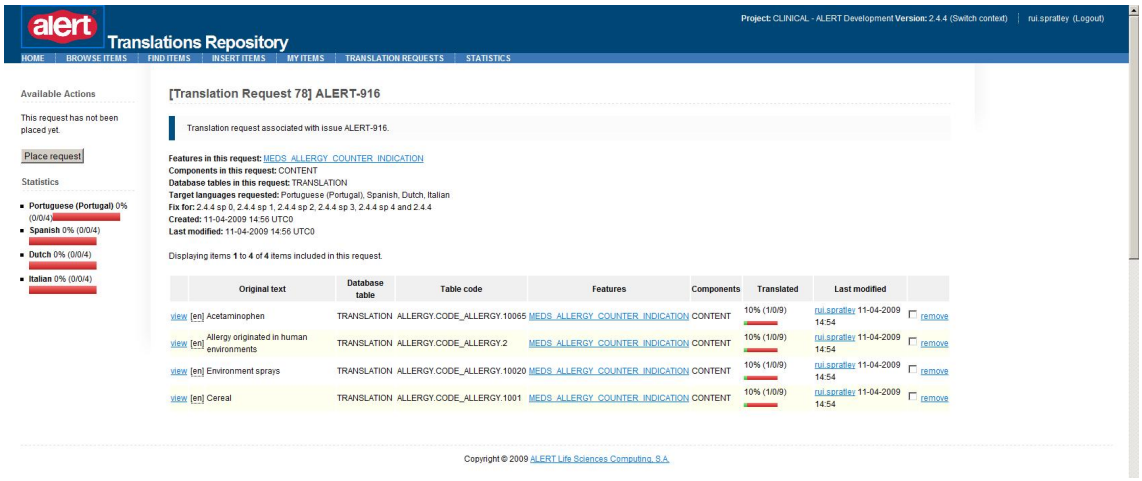

Figura B.2: Detalhes de um pedido de tradução.

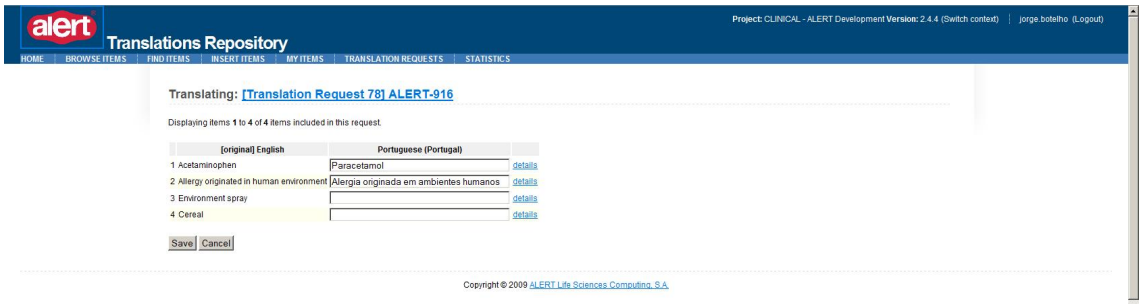

Figura B.3: Realizar um pedido de tradução directamente na aplicação.