FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Análise e Desenvolvimento de Módulos Funcionais para o ALERT® PHR

Ricardo Jorge Rodrigues Vieira

Relatório de Projecto
Mestrado Integrado em Engenharia Informática e Computação

Orientador: Prof. Doutor Ademar Manuel Teixeira de Aguiar

Junho de 2009

Análise e Desenvolvimento de Módulos Funcionais para o ALERT® PHR

Ricardo Jorge Rodrigues Vieira

Relatório de Projecto
Mestrado Integrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: João Francisco de Sousa Cardoso (Professor Auxiliar Convidado)

_____

Arguente: José Maria Amaral Fernandes (Professor Auxiliar Convidado)

Vogal: Ademar Manuel Teixeira de Aguiar (Professor Auxiliar)

29 de Junho de 2009

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 está sujeita a autorização prévia e por escrito da ALERT.

Resumo

O produto ALERT® PHR consiste numa aplicação web que pretende fornecer aos seus utilizadores um maior controlo e autonomia sobre os seus cuidados pessoais de saúde. Este centraliza a informação de saúde de cada utilizador e fornece funcionalidades que poderão conduzir a eventuais melhorias nos seus hábitos de saúde.

O intuito deste projecto consiste no desenvolvimento de novos módulos funcionais para o ALERT® PHR. Apresenta-se como um objectivo bastante aliciante uma vez que estes novos desenvolvimentos poderão contribuir significativamente para a melhoria da saúde dos diversos utilizadores do produto. O seu elevado número de utilizadores permite que todas as funcionalidades implementadas tenham uma grande visibilidade.

O desenvolvimento foi realizado de acordo com a infra-estrutra de desenvolvimento actual na empresa para o ALERT® PHR. A arquitectura lógica da aplicação encontra-se num modelo em três camadas (three-tier): uma camada que fica responsável pela interface da aplicação, uma que serve de middleware e inclui a lógica de segurança de sessões e autenticação e, por fim, uma que implementa toda a lógica de negócio, bem como o armazenamento dos dados. As tecnologias de desenvolvimento para cada uma delas, são, respectivamente, Flash, Java e Oracle DB. Os módulos funcionais a desenvolver no âmbito deste projecto inserem-se na camada de interface em Flash e são realizados na linguagem Flash ActionScript 2.0.

Antes de iniciar o desenvolvimento foram levantados todos os requisitos para os módulos funcionais requeridos. Foram documentados e ilustrados (com recurso a diagrama UML) todos os casos de uso para os actores correspondentes.

A implementação de todos os módulos funcionais, foi realizada após uma cuidada realização da sua arquitectura. Factores como a escalabilidade e performance foram cruciais na elaboração da mesma, o que levou à utilização de padrões de desenho de software sempre que possível. De modo a integrar com sucesso estes módulos funcionais no produto foi necessário compreender a sua organização e estrutura.

Todas as funcionalidades propostas foram implementadas com sucesso. Encontram-se, inclusive, já disponibilizadas aos seus utilizadores.

O desenvolvimento em Flash (com recurso a Flash AS 2.0) mostrou-se num desafio bastante interessante. Pois, apesar de ser bastante maleável, a sua biblioteca standard é limitada, possuindo poucas estruturas de dados já de raiz. O desenvolvimento nesta tecnologia torna-se bastante moroso e a realização de *debugging* bastante complicada. Actualmente, existem no mercado diversas alternativas tecnológicas mais actuais que proporcionam um melhor desenvolvimento de RIAs a todos os níveis, como por exemplo Adobe Flex e Microsoft Silverlight.

Abstract

The ALERT® PHR product is a web-based application that has the objective of providing its users a bigger control and autonomy over their personal health care data. This application centralizes information about each user's health and provides functionalities that may lead to improvements in their healthcare habits.

The purpose of this project is to develop new functional modules for the ALERT® PHR product. It presents itself as a challenging objective once these new developments may contribute in a significant way to the improvement of healthcare habits of the product's several users. Its high number of users allows that all its implemented functionalities have great visibility.

The development was made according to the present development infra-structure of the ALERT® PHR product in the company. The logic architecture can be described as a three-tier model: the first tier is responsible of handling the applications' user-interface, the second one acts like a middleware and has embbeded the session security and authentication logic while the third one implements all the business logic as well as data storage. The development platforms are respectively Flash, Java and Oracle DB. The funcitonal modules to develop are located within the user-interface tier implemented using the Flash platform and are made using Flash ActionScript 2.0 programming language.

Before starting the development, all project requirements were defined. All the use cases for the corresponding actors were illustrated and documented.

All functional modules development was made after a careful designing of its architecture. Factors as performance and scalability were crucial in its elaboration which led to the use of software design patterns when possible. It was necessary to understand its organization and structure so these modules could be successfully integrated.

The proposed functionalities were sucsessfully implemented, and were already made available to its users.

Developing these function modules using Flash AS 2.0 as turned out to be quite an interesting challenge, because despite being pretty flexible, its standard library is limited because it has only a few pre-defined data structures. Developing an application using this technology takes a reasonable amount of time and makes debugging a difficult task. Nowadays there are several market alternative technologies that are more up to date and provide a better RIA development in every possible aspects. Some examples of these technologies are Adobe Flex and Microsoft Silverlight.

Agradecimentos

Gostaria de agradecer ao Prof. Doutor Ademar Aguiar pela orientação que me forneceu ao longo do projecto e pelo seu constante apoio durante a realização do mesmo.

Agradeço a toda a minha equipa do ALERT® PHR pelo seu companheirismo e amabilidade com que todos me receberam e sempre me trataram. Bem como, pelo apoio técnico fornecido que sempre se encontraram disponíveis para me dar. Em especial, ao Vítor Monteiro e Rui Neves pelo seu incansável apoio e paciência.

Agradeço também aos meus amigos mais chegados (eles sabem quem são) pelos bons momentos que sempre passamos em conjunto e pela força que me transmitiram nos momentos mais cruciais.

Por último, mas nunca menos importante, agradeço à minha família pelo seu valiosíssimo apoio, paciência e dedicação, sem os quais a realização desta tese não teria sido possível.

Muito obrigado.

Ricardo Jorge Rodrigues Vieira

Índice

1 Introdução 9
1.1 Empresa 9
1.2 Produtos ALERT® 11
1.2.1 ALERT® PHR 11
1.3 Motivação e Objectivos 12
1.4 Estrutura do Relatório 12
2 Infra-Estrutura de Desenvolvimento 14
2.1 Arquitectura de Aplicação 14
2.2 Tecnologias Utilizadas 15
2.2.1 Adobe Flash 8 15
2.2.2 Apache Tomcat 16
2.2.3 Base de Dados Oracle 16
2.3 Ferramentas de Apoio ao Desenvolvimento 16
2.3.1 Eclipse IDE 16
2.3.2 X-Ray 17
2.3.3 Service Capture 17
2.4 Controlo de Qualidade 18
3 Descrição do Problema 19
3.1 Especificação de Requisitos 20
3.1.1 Visão Geral 20
3.1.1.1 Actores 21
3.1.2 Componentes 22
3.1.2.1 Acerca de uma Ferramenta 23
3.1.2.2 Título Dinâmico 23
3.1.2.3 Menu de Calculadora 24
3.1.2.4 Calendário 24
3.1.3 Funcionalidades 25
3.1.3.1 Ajuda Contextualizada 25
3.1.3.2 Notas de Lançamento 27
3.1.3.3 Calculadoras 28

4 Desenvolvimento 31
4.1 Conceitos Fundamentais para o desenvolvimento em Flash 8 31
4.2 Arquitectura e Implementação 33
4.2.1 Componentes 33
4.2.1.1 Desenvolvimento de UIComponents em Flash 8 33
4.2.1.2 Acerca desta Ferramenta 34
4.2.1.3 Título Dinâmico 35
4.2.1.4 Menu da Calculadora 36
4.2.1.5 Calendário 37
4.2.2 Funcionalidades 42
4.2.2.1 Viewer 43
4.2.2.1.1 Ajuda Contextualizada 44
4.2.2.1.2 Calculadoras 46
4.2.2.2 Screen 48
4.2.2.2.1 Notas de Lançamento 50
5 Conclusões e Perspectivas Futuras 53
6 Bibliografia 54

Lista de Figuras

Figura 1.1: Grupo de Empresas ALERT 10
Figura 2.1: Arquitectura Lógica do ALERT® PHR 15
Figura 3.1: Diagrama de visão geral de casos de uso 21
Figura 3.2: Actor Utilizador Autenticado 21
Figura 3.3: Diagrama de casos de uso do pacote "Componentes" 22
Figura 3.4: Diagrama de casos de uso da "Ajuda Contextualizada" 26
Figura 3.5Diagrama de estado da "Ajuda Contextualizada" 26
Figura 3.6: Diagrama de casos de uso das "Notas de Lançamento" 27
Figura 3.7: Diagrama de estado das "Notas de Lançamento" 28
Figura 3.8: Diagrama de casos de uso das "Calculadoras" 29
Figura 3.9: Diagrama de estado da funcionalidade "Calculadoras" 30
Figura 4.1: Padrão de desenho Composite aplicado ao MovieClip 32
Figura 4.2: Diagrama de classes do componente "Acerca desta Ferramenta" 34
Figura 4.3: Dois estados possíveis para o componente "Título Dinâmico" 35
Figura 4.4: Diagrama de classes do componente "Título Dinâmico" 36
Figura 4.5: Área de visibilidade do "Título Dinâmico" 36
Figura 4.6: Movimento de aperecimento do subtítlo do componente "Titulo Dinâmico" 36
Figura 4.7: Exemplos dos dois estados possíveis do componente "Menu da Calculadora" 37
Figura 4.8: Componente do Calendário 37
Figura 4.9: Componente do "Calendário" a permitir apenas a inserção de hora 38
Figura 4.10: Pop-up do componente "Calendário" que permite a selecção de uma data 39
Figura 4.11: Diagrama de classes do componente "Calendário" 39
Figura 4.12: Informação prévia necessária para correr o algoritmo de Doomsday 40
Figura 4.13: Zona desenhada pela classe Calendar do componente "Calendário" 41
Figura 4.14: Zonas principais constituintes do ALERT® PHR 43
Figura 4.15: Diagrama de classes do Viewer 44
Figura 4.16: Exemplo da funcionalidade "Ajuda Contextualizada" 45
Figura 4.17: Diagrama de classe da funcionalidade "Ajuda Contextualizada" 45
Figura 4.18: Exemplo da funcionalidade "Calculadoras" 46

Figura 4.19: Diagrama de classes da funcionalidade "Calculadoras" 47
Figura 4.20: Zona de selecção da categoria da calculadora a usar 47
Figura 4.21: Legendas das diferentes secções constituintes da calculadora 48
Figura 4.22: Botões e Deepnavs do ALERT® PHR 49
Figura 4.23: Diagrama de classes do "Screen" do ALERT® PHR 50
Figura 4.24: Diagrama de classes da funcionalidade "Notas de Lançamento" 51
Figura 4.25: Pop-up de notificação de lançamento de nova versão/fix do ALERT® PHR 51
Figura 4.26: Legenda das diferentes secções que constituem a funcionalidade "Notas de Lançamento" 52

Abreviaturas e Símbolos

AMF	ActionScript Message Format
AS	ActionScript
EHR	Electronic Health Record
FDT	Flash Development Tool
HTTP	Hypertext Transfer Protocol
IDE	Integrated Development Environment
OOP	Object-Oriented Programming
PHR	Personal Health Record
RIA	Rich Internet Application
SDK	Software Development Kit
SGBD	Sistema de Gestão de Base de Dados
SVN	Subversion
SWF	Shockwave Flash
UI	User Interface
UML	Unified Modeling Language
WWW	World Wide Web
XML	Extensible Markup Language

1 Introdução

Este capítulo destina-se a apresentar a empresa, assim como o produto sobre o qual o projecto foi realizado. É descrita a motivação que levou ao seu desenvolvimento e os seus objectivos. Por fim, é definida a estrutura do relatório.

1.1 Empresa

O projecto em causa foi efectuado na empresa ALERT Life Sciences Computing, S.A.

O Grupo de Empresas ALERT dedica-se ao desenvolvimento, distribuição e implementação do software de saúde ALERT®, concebido para criar ambientes clínicos sem papel.

A ALERT começou a sua actividade em Dezembro de 1999. Anteriormente designada MNI - Médicos Na Internet, a ALERT Life Sciences Computing, S.A. é a empresa mãe do Grupo de Empresas ALERT. Está inteiramente dedicada ao desenvolvimento, distribuição e implementação do software clínico ALERT®.

Actualmente, o Grupo conta hoje com uma equipa multidisciplinar de clínicos, designers, arquitectos, engenheiros, matemáticos e gestores. O grupo ALERT inclui empresas em sete localizações diferentes: Portugal, Estados Unidos da América, Espanha, Holanda, Singapura, Brasil e Reino Unido.

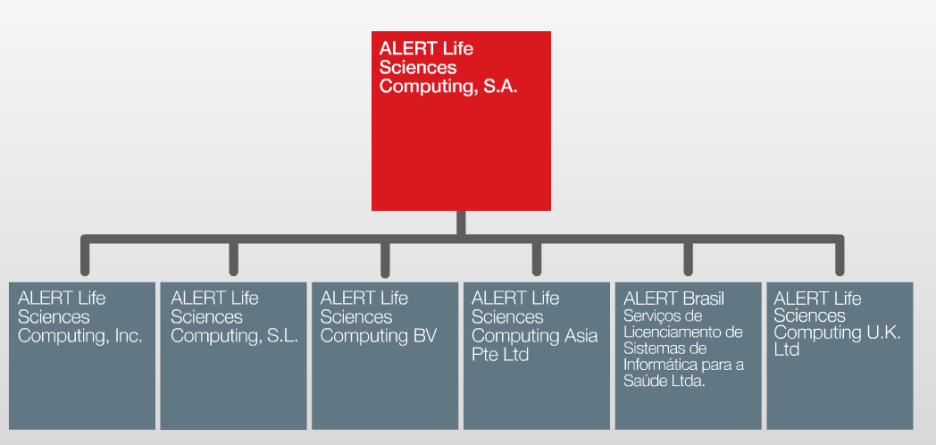

Figura 1.1: Grupo de Empresas ALERT

A missão da ALERT consiste em melhorar a saúde e prolongar a vida, alcançar rentabilidade para benefício da sociedade e inspirar outros para a excelência, através do nosso exemplo. Os seus valores estão representados na excelência, competência, transparência, generosidade e amor pela vida em todas as suas formas.

A ALERT Life Sciences Computing, S.A. integra a Rede de PME's Inovadoras da COTEC e a sua metodologia de formação foi seleccionada pelo Fundo Social Europeu para integrar duas publicações que ilustram casos de sucesso.

A empresa é Platinum Corporate Member da HIMSS (Healthcare Information and Management Systems Society) e tem participado nas iniciativas IHE (Integrating the Healthcare Enterprise). A IHE permite acelerar a adopção de Processos Clínicos Electrónicos, já que promove uma melhor troca de informação entre sistemas de saúde. O objectivo desta iniciativa consiste em melhorar a qualidade, eficiência e segurança dos cuidados clínicos através da disponibilização de informação clínica relevante tanto a pacientes como a prestadores de cuidados de saúde devidamente autorizados. Até à data, a ALERT publicou 12 Certificados de Interoperabilidade IHE.

A empresa apresenta um já longo historial de prémios e distinções. Incluíndo, um da própria COTEC entregue pelas mãos do Presidente da República Aníbal Cavaco Silva, o Prémio PME Inovação COTEC – BPI. Durante a cerimónia o Juiz destacou o carácter inovador e a sustentabilidade da solução proposta através do sistema ALERT – baseados em conhecimentos e capacidades de organização - e realçou o contributo da empresa para a criação de novos mercados, evidenciados pela sua taxa de crescimento e internacionalização, num sector de enorme relevo que é o da saúde.

Em 2008, a ALERT apresentou um volume de negócios superior a 35 milhões de euros (51 milhões de dólares) , o que mantém a tendência de aumentar rapidamente as suas vendas todos os anos. Estes resultados reflectem o investimento considerável da empresa no desenvolvimento e comercialização de produtos de qualidade.

O ALERT® já foi adoptado em Portugal, Espanha, Itália, Holanda, Inglaterra, Estados Unidos, Brasil e Malásia [AOL09].

1.2 Produtos ALERT®

A suite ALERT® disponibiliza soluções não só para todo o ambiente hospitalar, como também directamente para o cidadão.

Em termos de software clínico, podemos encontrar produtos para várias necessidades distintas que satisfazem clínicas, centros de saúde e/ou hospitais. Portanto, existem soluções para serviços de urgência hospitalares, consulta externa, internamento, gestão de bloco operatório, entre outros.

O produto relativo a este projecto, ALERT® Personal Health Record (a partir de agora designado por ALERT® PHR), insere-se mais especificamente na área do cidadão [AOL09].

1.2.1 ALERT® PHR

O ALERT® PHR consiste numa aplicação que permite ao cidadão registar os seus dados clínicos através da internet.

É um produto que dota os seus utilizadores da capacidade de organizar, manter e gerir eficazmente o seu registo clínico electrónico, através da disponibilização de todos os seus dados de saúde num único local. Deste modo, permite que os seus utilizadores participem activamente na documentação do seu estado de saúde, inserindo e editando informações relativas aos seus problemas, alergias, resultados de análises, medicação e vacinação. Para o efeito, são incluídas nomenclaturas internacionais de documentação e codificação de diagnósticos para que os registos de saúde sejam interpretáveis segundo uma norma. Actualmente, são disponibilizados conteúdos clínicos para os mercados de Portugal, Estados Unidos da América e Brasil.

O ALERT® PHR também disponibiliza ferramentas que permitem ao utilizador a monitorização e avaliação de diversos aspectos de saúde e qualidade de vida, como hábitos alimentares, de sono, tabágicos, entre outros.

Em suma, o ALERT® PHR é um software que oferece uma solução completa que permite aos seus utilizadores, aceder, gerir e contribuir para o seu próprio registo de saúde electrónico (Electronic Health Record, EHR). Um dos futuros objectivos desta aplicação é a interacção com os restantes produtos ALERT® e outros softwares clínicos, de modo a facilitar a partilha de dados entre os cidadãos e profissionais de saúde. Como actualmente não existe nenhuma solução deste género no mercado, esta evolução pode consistir numa grande mais-valia para a empresa e numa funcionalidade *ground-breaking* [AOL09].

1.3 Motivação e Objectivos

A escolha deste projecto deveu-se principalmente ao objecto de trabalho em questão, o ALERT® PHR.

Sendo o ALERT® PHR uma aplicação para gestão do perfil clinico do cidadão com várias dezenas de milhares de licenças contractualizadas, existe logo à partida um conjunto de factores motivantes:

- **Contributo Social:** Ao fazer parte do desenvolvimento do ALERT® PHR o programador está a criar uma solução de software cujo objectivo final é ajudar o utilizador a obter mais consciencia sobre os seus problemas, cuidados de saúde e soluções a adoptar. Mesmo que o produto tenha impacto numa percentagem reduzida de utilizadores, esses poderão antecipar e/ou resolver problemas de saúde que em última análise podem ser o ponto determinante para uma vida saúdavel.

- **Produto vanguardista na sua área:** Há poucos anos atrás havia pouco conhecimento do que era um Personal Health Record e a sua necessidade não era reconhecida. Hoje em dia os governos dos paises desenvolvidos vêm os PHR's como uma necessidade de primeira ordem para os seus habitantes [CNN09]. Empresas como a Microsoft com o HealthVault e a Google com o Google Health já apostaram neste ramo da saúde [MHV09] [GH09].

- **Responsabilidade:** Sendo o ALERT® PHR um produto que neste momento já tem contratualizadas dezenas de milhares de licenças que podem porventura aumentar num futuro próximo, o desenvolvimento da aplicação é da maior responsabilidade para os seus intervenientes. Não é tolerada qualquer falha, pois esta poderá por em causa o bem estar dos seus utilizadores.

O objectivo deste projecto de mestrado é evoluir o ALERT® PHR da forma mais eficiente, modular e produtiva de forma a tentar melhorar o bem-estar dos seus utilizadores através de um programa usável, rápido e coerente.

1.4 Estrutura do Relatório

Este documento pretende descrever todas as etapas ultrapassadas para a realização deste projecto.

No capítulo actual, é apresentado a empresa e o produto no qual este projecto se insere. São descritos os factores motivadores que levaram à realização do mesmo e os seus objectivos.

No próximo capítulo, encontra-se definida a infra-estrutura de desenvolvimento em vigor na ALERT para o produto ALERT® PHR, as tecnologias utilizadas no seu

desenvolvimento, ferramentas de apoio ao processo e o modo como é realizado o controlo de qualidade sobre as implementações efectuadas.

No terceiro capítulo encontra-se toda a documentação relativa ao levantamento de requisitos realizado para todas as funcionalidades requisitadas, incluido os diagramas de caso de uso com respectiva explicação e seus diagramas de estado.

Seguidamente, no quarto capítulo, é descrita toda a arquitectura e implementação dos módulos funcionais desenvolvidos. São também apresentados aspectos fundamentais para se compreender como se processa o desenvolvimento em Flash 8, explicado como se extende e evolui a framework do ALERT® PHR e, ainda, como e onde os módulos funcionais devem ser integrados no produto.

Por fim, no último capítulo, é realizada uma análise crítica a todas as fases de realização do projecto. São apresentadas as conclusões retiradas do desenvolvimento deste e é feita uma revisão ao cumprimento dos objectivos inicialmente propostos. A concluir, é realizada uma consideração sobre as perspectivas futuras para o produto ALERT® PHR.

2 Infra-Estrutura de Desenvolvimento

Nesta secção é descrita toda a infra-estrutura de desenvolvimento utilizada na ALERT, mais especificamente a do produto ALERT® PHR. Assim, apresenta-se a arquitectura da aplicação, as tecnologias utilizadas, ferramentas de apoio ao desenvolvimento e o controlo de qualidade realizado às soluções implementadas.

2.1 Arquitectura de Aplicação

Uma arquitectura multi-camadas (n-tier) consiste numa arquitectura de cliente-servidor estruturada por camadas, onde cada uma cumpre determinadas tarefas. A tecnologia de implementação de cada pode diferir para melhor se adaptar às suas necessidades. Desde que respeitem a mesma interface, uma camada pode ser substituída por outra sem afectar o normal funcionamento das restantes. Assim, uma das suas maiores vantagens consiste na independência do desenvolvimento entre camadas diferentes, podendo ser realizado de modo completamente assíncrono.

O modelo mais comum deste tipo de arquitectura trata-se do *three-tier*. Neste modelo, existe uma camada responsável pela interface da aplicação, outra pela lógica de negócio e uma última pela base de dados.

No ALERT® PHR, é usada a arquitectura three-tier, porém, não segundo o modelo convencional. Aqui as camadas e as suas respectivas funções são as seguintes:

- **Camada em Adobe Flash 8** – responsável pela interface gráfica;
- **Camada em Java com servidor aplicacional Tomcat** – serve de intermediário para a comunicação entre a camada de interface e de base de dados e inclui a lógica de segurança de sessões e autenticação. A necessidade da existência desta camada é justificada pelo facto de o Flash não ser capaz de comunicar directamente com a base de dados;
- **Camada de Base de Dados em Oracle** – implementa toda a lógica de negócio da aplicação, bem como o armazenamento dos dados.

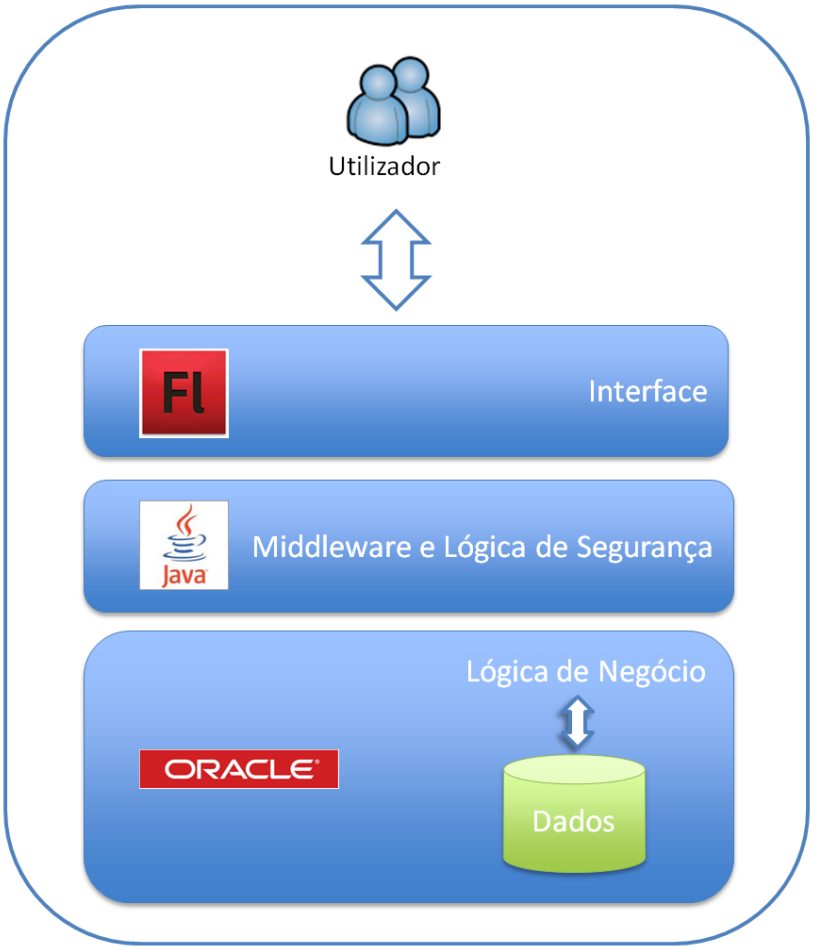

Figura 2.1: Arquitectura Lógica do ALERT® PHR

2.2 Tecnologias Utilizadas

2.2.1 Adobe Flash 8

O Flash 8 trata-se de uma plataforma multimédia desenvolvida pela Macromedia. Actualmente, já na versão 10, pertence à Adobe Systems que está agora a cargo do seu desenvolvimento e distribuição.

Desde a sua introdução em 1996 que se tornou num método bastante popular para adicionar animação e interacção às páginas web. A sua aplicabilidade é vasta e é comum verificar a sua utilização na realização de animações, publicidade, integração de aúdio, vídeo ou no desenvolvimento de rich internet applications.

O Flash é capaz de manipular imagens (vectorias e bitmap) e suporta streaming bidireccional de aúdio e vídeo. Possui também uma linguagem de scripting, o ActionScript, especialmente útil para o auxílio na criação de jogos web e rich internet applications [AFL09].

Consiste numa tecnologia bastante massificada, não só pela grande quantidade de utilizadores que detém, mas também pela variedade de dispositivos e de software existente capazes de reproduzir o seu conteúdo. Segundo a própria Adobe, o Flash Player encontra-se instalado em 99.3% dos computadores com ligação à internet [ADB09].

2.2.2 Apache Tomcat

O Apache Tomcat é uma implementação open source das tecnologias Java Servlet e JavaServerPages (JSP) da Sun Microsystems. É desenvolvido pela Apache Software Foundation (ASF) e disponibiliza um ambiente de servidor web HTTP que permite correr código Java.

O Tomcat não deve ser confundido com o servidor web Apache, desenvolvido pela mesma fundação. Este último, trata-se de uma implementação de um servidor web HTTP. Possuem finalidades diferentes, portanto nem sequer são disponibilizados conjuntamente num só pacote [WATC09].

A configuração e gestão do Tomcat é realizada através de ferramentas que acompanham a sua distribuição. Opcionalmente, este tipo de operações pode ser também realizado através da edição de ficheiros XML [ATC09].

2.2.3 Base de Dados Oracle

A base de dados Oracle (Oracle Database) consiste num sistema de gestão de base de dados relacional, produzido e comercializado pela Oracle Corporation. Actualmente, é a mais usada a nível mundial com fins comerciais [ODB09].

A base de dados Oracle é uma das mais rápidas, se não mesmo a mais rápida do mercado [ORP09]. Além das capacidades de armazenamento e pesquisa de dados que oferece, suporta implementações de lógica de negócio. É também caracterizada pela sua fiabilidade e segurança dos dados [ORC09].

2.3 Ferramentas de Apoio ao Desenvolvimento

2.3.1 Eclipse IDE

O Eclipse é uma plataforma de desenvolvimento de software multi-linguagem que inclui um ambiente de desenvolvimento integrado (IDE) e um sistema de plug-ins que lhe permite extender as suas funcionalidades. Foi escrito maioritariamente em Java e

pode ser usado para desenvolver nessa linguagem, ou, através do recurso aos referidos *plug-ins*, em várias outras. O seu uso é gratuito e trata-se de software open source [ECLPS09].

Neste projecto foram usados dois *plug-ins*:

- **Flash Development Tools Plug-in (FDT)** – Em conjunto com o eclipse, este *plug-in* permite disponibilizar um conjunto de funcionalidades para auxiliar o desenvolvimento em Flash. Alguns exemplos são:
  - Preenchimento automático de código;
  - Detecção de erros em tempo real;
  - Organização de imports;
  - Janela de visualização de ficheiros Flash (SWF);
  - Sugestões de correcção de erros [FDT09].

- **Subversion SVN** – É um sistema de controlo de versões que permite que o software a ser desenvolvido seja realizado de modo incremental. Mantém um histórico de todas as alterações realizadas sobre os ficheiros que constituem o projecto e oferece ferramentas que permitem a integração do trabalho realizado por todos os colaboradores envolvidos [SBCPS09].

2.3.2 X-Ray

O X-Ray consiste numa aplicação Flash que permite auxiliar o debugging nessa mesma tecnologia.

Entre as suas funcionalidades, destaca-se a capacidade de apresentação de mensagens de debug e a de fornecer uma árvore de objectos de uma aplicação Flash, listar todas as suas propriedades e, em tempo real, alterar os seus valores vendo de imediato o efeito produzido na respectiva aplicação.

Em suma, esta ferramenta possibilita que o tempo perdido para detectar a causa de um bug diminua [XRAY09].

2.3.3 Service Capture

O Service Capture é uma ferramenta capaz de capturar todo o tráfego HTTP realizado. Está especialmente desenhado para ajudar os programadores de Rich Internet Application (RIA) no debugging, análise e teste às suas aplicações.

É uma aplicação capaz de decompor toda a informação relativa ao tráfego HTTP numa estrutura de fácil leitura para o programador. Deste modo, fornece informação detalhada acerca da chamada a métodos do servidor aplicacional – argumentos de entrada e saída e retorno obtido.

Para cada pedido, também é possível consultar o código HTTP devolvido e a duração do tempo até a resposta ter sido obtida.

Esta ferramenta trata-se de uma mais valia para medir a performance do produto ou identificar quando um erro é provocado pela interface, servidor aplicacional ou pela base de dados [SCP09].

2.4 Controlo de Qualidade

O controlo de qualidade do produto é realizado por uma equipa que assegura que todas as funcionalidades desenvolvidas correspondem ao esperado – em termos funcionais, de usabilidade e de desempenho. Qualquer situação insatisfatória (por exemplo, um bug encontrado) deve ser por ela reportada, assim como eventuais sugestões para a melhora do produto.

O registo de todas estas situações é realizado numa aplicação web de nome JIRA. Esta permite prioritizar, atribuir, acompanhar, relatar e auditar "issues". Uma "issue" pode consistir desde um bug de software, a *tickets* de *help-desk*, até tarefas de um projecto ou pedidos de alterações. O JIRA é usado no desenvolvimento de software, gestão de projectos e *helpdesks* e foi concebido com o objectivo principal de auxiliar os *developers* e gestores no cumprimento de tarefas.

O JIRA oferece uma grande capacidade de customização, o que o torna fácil de adaptar ao processo de negócio de uma empresa [JIRA09].

3 Descrição do Problema

O produto ALERT® PHR pretende fornecer aos seus utilizadores um maior controlo e autonomia sobre os seus cuidados pessoais de saúde. Este centraliza a informação da saúde de cada utilizador e fornece funcionalidades que poderão conduzir a eventuais melhorias nos seus hábitos de saúde. Deste modo, possibilita-se uma gestão eficiente do processo clínico de um qualquer cidadão sem que este tenha necessidade de ter qualquer conhecimento clínico. Assim, espera-se que a aplicação tenha uma influência positiva na qualidade de vida de todos os seus utilizadores.

No âmbito da produção por um produto cada vez melhor, foram valorizadas as questões a nível de funcionalidade e usabilidade, mas igualmente melhoramentos ao nível da Framework técnica em que o ALERT® PHR está construído. A evolução da Framework do ALERT® PHR a nível da camada de interface com o utilizador era, então, igualmente, um dos objectivos deste projecto.

Devido à necessidade mais pertinente de evolução da Framework sobre a camada de interface gráfica, foi aí que todo o trabalho, no seguimento do projecto, foi efectuado.

Os melhoramentos à Framework incluídos como parte do projecto foram os seguintes:

- **Acerca desta Ferramenta** – disponibiliza informação explicativa acerca de uma funcionalidade;
- **Título Dinâmico** – oferece um título e um subtítulo para uma secção. O subtítulo poderá estar ou não visível;
- **Menu de Calculadora** – menu intuitivo que permite escolher várias opções a realizar sobre uma calculadora;
- **Calendário** – permite a especificação de uma data e hora.

A nível de funcionalidades para o produto:

- **Calculadoras**:
  - Gestão do peso – permite calcular o índice de massa corporal;
  - Estilo de Vida – calcula o número de calorias diárias recomendadas;
  - Saúde da Mulher – estima uma data para parto;

- **Ajuda Contextualizada** – oferece ajuda acerca da funcionalidade correntemente em uso pelo utilizador;
- **Notas de Lançamento** – disponibiliza toda a informação relativa a uma nova versão do produto;

Uma descrição mais pormenorizada de cada uma das funcionalidades será efectuada mais à frente no presente documento.

3.1 Especificação de Requisitos

Em seguida, será apresentada a especificação de requisitos das funcionalidades desenvolvidas no âmbito do projecto.

3.1.1 Visão Geral

As funcionalidades e componentes desenvolvidos encontram-se agrupados por pacotes. Deste modo, no pacote denominado “Componentes” encontram-se os casos de uso respectivos a todos os componentes desenvolvidos para a framework do produto;

Os pacotes Notas de Lançamento, Ajuda Contextualizada e Calculadoras dizem respeito às funcionalidades implementadas de nome equivalente.

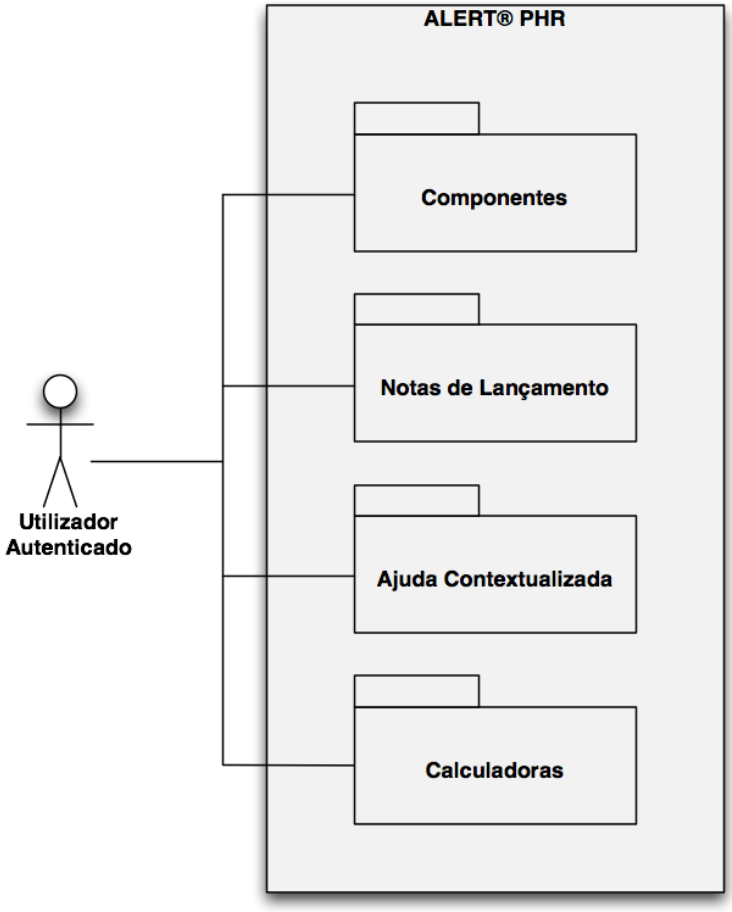

Figura 3.1: Diagrama de visão geral de casos de uso

3.1.1.1 Actores

O único actor existente para o ALERT® PHR é efectivamente o utilizador autenticado dado que não faz sentido que um utilizador anónimo aceda a uma informação que poderá roçar a confidencialidade. Este pode usufruir de todos os casos de uso presentes nestes pacotes.

Figura 3.2: Actor Utilizador Autenticado

3.1.2 Componentes

Neste pacote é possível encontrar os casos de uso para os componentes a serem desenvolvidos para a framework do produto. Justifica-se que apresentem casos de uso, uma vez que se tratam de componentes com interface gráfica, cujo estado é, pelo menos em parte, destinado a ser controlado pelo utilizador.

Assim, este pacote engloba sub-pacotes para cada cada componente desenvolvido:

- Acerca de uma Ferramenta;
- Título Dinâmico;
- Menu de Calculadora;
- Calendário.

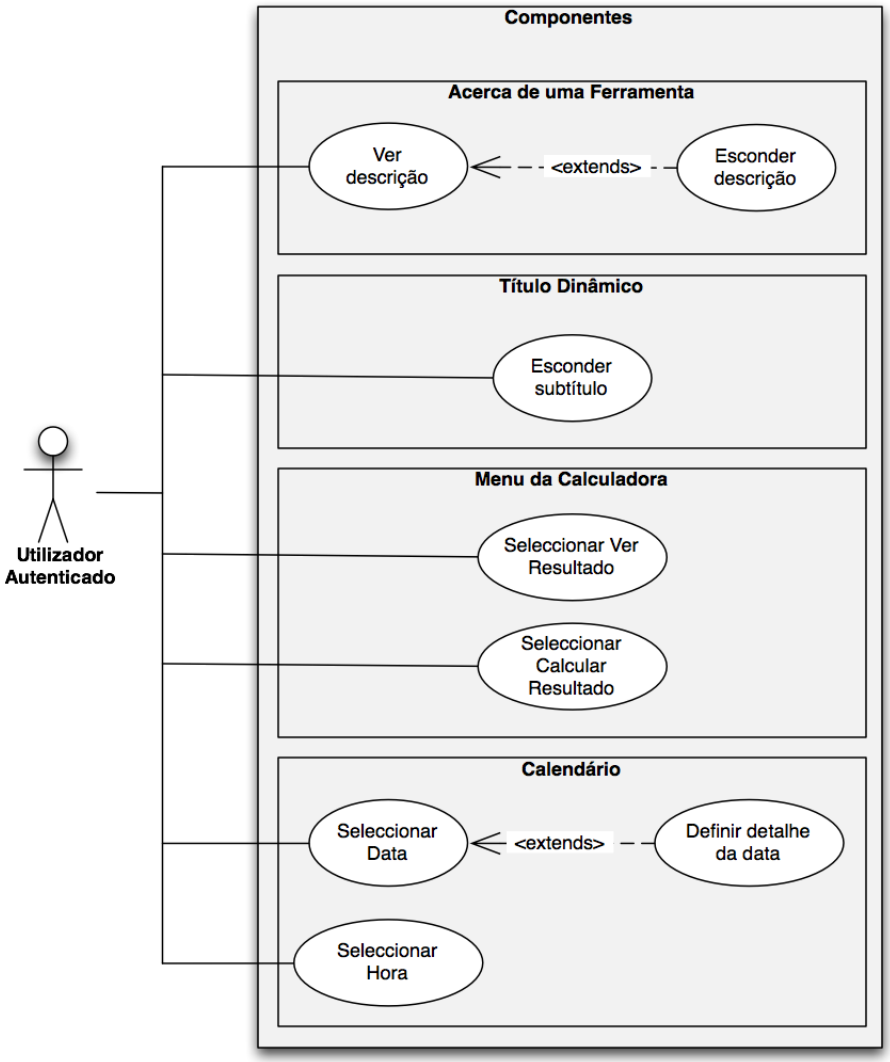

Figura 3.3: Diagrama de casos de uso do pacote "Componentes"

3.1.2.1 Acerca de uma Ferramenta

A finalidade deste componente consiste em disponibilizar informação adicional descritiva acerca de uma ferramenta/funcionalidade do produto.

Este deverá ser constituído por um título e conter uma secção para a descrição da ferramenta a que se refere. Esta secção surge escondida por defeito mas poderá ser revelada através de um clique no título. Este componente também inclui uma imagem de fundo alusiva ao texto.

Descrição dos Casos de Uso

Ver descrição

Quando é realizado um clique sobre o título deste componente, surge imediatamente em baixo, um texto descritivo acompanhado por uma imagem de fundo. Para esta acção ser possível é necessário que a descrição esteja inicialmente escondida (estado inicial do componente).

Esconder descrição

Semelhante ao caso de uso "Ver descrição". No entanto, neste caso quando é realizado um clique sobre o título deste componente, esconde-se imediatamente o texto descritivo e respectiva imagem de fundo. Ao contrário do caso de uso referido anteriormente, para este se realizar é necessário que o descritivo esteja visível.

3.1.2.2 Título Dinâmico

Destina-se a secções do produto onde exista a necessidade de existir uma secção que possua várias sub-secções e que ofereça a possibilidade de navegar para elas.

Assim, se o utilizador se encontra na secção principal, apenas se encontra vísivel o título dessa secção. Quando o utilizador navega para uma das sub-secções, surge, ao lado do título principal, o título da sub-secção. Quando neste estado, é possível retroceder recorrendo ao clique sobre este componente. Deverá ser possível controlar o estado de visibilidade do subtítulo, bem como requisitar notificações para quando o utilizador interagir com o título. Este comportamento gráfico, é adoptado em situações em que o titulo serve como navegador entre secções.

Descrição dos Casos de Uso

O componente ”Título Dinâmico” apenas contem um caso de uso, “Esconder Subtitulo”. Este acontece quando o utilizador clica sobre o componente e o subtilulo é consequentemente escondido. A única pré-condição para verificação deste caso de uso é o subtítulo estar originalmente visível.

3.1.2.3 Menu de Calculadora

Este componente surgiu para colmatar uma necessidade específica da funcionalidade “Calculadoras”. Deve possibilitar um rápido e intuitivo acesso ao cálculo de um novo resultado, bem como à consulta do último obtido. Deverá, também, apresentar informação sumária acerca do mesmo.

Descrição dos Casos de Uso

Seleccionar Calcular Resultados

Possibilita a navegação para o formulário de calculo de uma determinada calculadora. Este formulário pode variar conforme o resultado que se pretende embora por vezes contenha campos de entrada semelhantes como o peso e a altura. Não existem pré-condições para este caso de uso.

Seleccionar Ver Resultados

Possibilita a navegação para a zona de consulta do último resultado obtido. Para este caso de uso ser válido é necessário que previamente tenha sido efectuado pelo menos um calculo com sucesso no resultado.

3.1.2.4 Calendário

O produto já possui um componente que permite a inserção de data e horas. No entanto, a sua usabilidade e bastante baixa e, até, desconfortável para o utilizador. Pois o utilizador deve inserir os valores prentendidos num conjunto de caixa de textos.

Neste novo componente, deve ser possível ao utilizador escolher a data pretendida através de um Calendário. O utilizador terá a possibilidade de especificar o detalhe que pretende para a mesma (só ano, ano e mês ou dia, ano e mês). No entanto, poderá, nos casos que se justifique, obrigar-se o utilizador a escolher forçosamente uma data completa. É, ainda, também possível obrigar o utilizador a escolher uma data passada ou futura.

Este componente poderá ser configurado para permitir escolher apenas datas ou horas, mas também datas e horas.

Uma vez que se pretende que este componente substitua o anterior, a interface de controlo deverá ser idêntica. Isto, de modo a permitir que a substituição seja realizada de um modo transparente nas diversas funcionalidades que necessitam de inserção de datas.

Descrição dos Casos de Uso

Seleccionar Data

Permite ao utilizador escolher uma data pela selecção de um ano ou ano/mes ou ano/mês/dia conforme o detalhe da data seleccionado. Para tal acontecer basta apenas que o componente esteja configurado de modo a permitir a especificação da data.

Definir Detalhe da Data

Consoante a alteração do detalhe da data o utilizador poderá optar por preencher ano, ano e mês ou ano, mês e dia. Esta função do componente só está disponível se préviamente for definida a configuração que permite disponibilizar a granularidade da data assim como a possibilidade de preencher uma data parcial).

Seleccionar Hora

Permite especificar uma hora até uma granularidade de cinco minutos desde que o componente esteja previamente configurado para permitir a especificação da hora.

3.1.3 Funcionalidades

Neste pacote são descritos os requisitos funcionais para as funcionalidades desenvolvidas para o produto.

3.1.3.1 Ajuda Contextualizada

O ALERT® PHR pretende ser um produto intuitivo de usar, com uma curva de aprendizagem muito curta e sem necessidade de qualquer tipo de formação por parte dos seus utilizadores.

É neste contexto que se insere esta funcionalidade. Pretende esclarecer todas as dúvidas que possam surgir aos utilizadores acerca de qualquer funcionalidade do produto. O seu conteudo é automaticamente adaptado ao menu que se encontra seleccionado a cada momento.

A informação apresentada deverá conter uma descrição geral do menu, acompanhada por informações mais pormenorizadas organizadas por categorias hierárquicas.

Descrição dos Casos de Uso

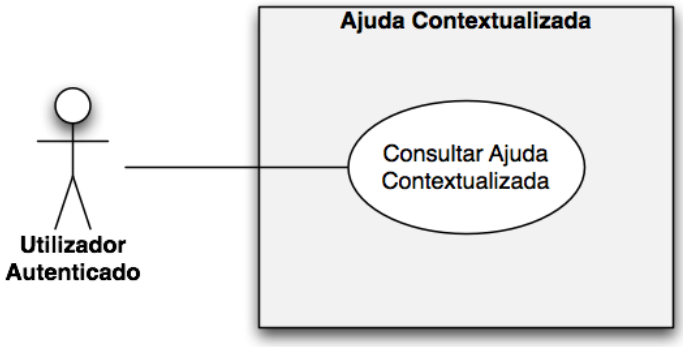

Figura 3.4: Diagrama de casos de uso da "Ajuda Contextualizada"

O caso de uso “Consultar Ajuda Contextualizada” permite a visualização dos textos de ajuda disponíveis para a funcionalidade correntemente em uso. Para tal basta que estejam disponiveis os conteudos que são apresentados ao utilizador neste componente.

Diagrama de Estados

No seguinte diagrama de estados é apresentada a navegação nesta funcionalidade. De notar que a consulta das diversas categorias é realizada sem necessidade de navegação para outro local.

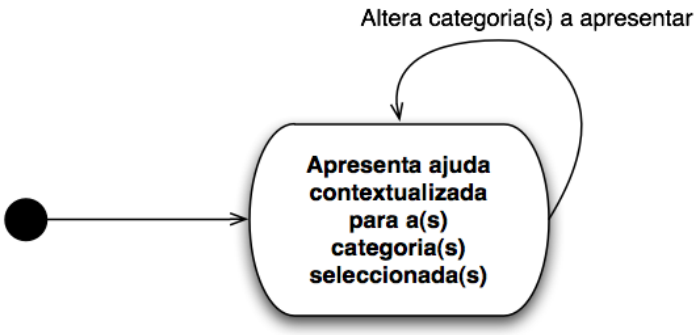

Figura 3.5Diagrama de estado da "Ajuda Contextualizada"

3.1.3.2 Notas de Lançamento

Uma vez que o produto se encontra em constante evolução, é normal que surjam novos melhoramentos e correcções. Este tipo de informação é de todo o interesse do utilizador, por isso é útil que exista um modo fácil de consultar esta informação e que esta se encontre sempre actualizada.

Assim, as actualizações no produto podem surgir sob duas formas distintas:

- Versão - Além de aglomerar diversas novas funcionalidades, pode também conter correcções/melhoramentos realizados a funcionalidades já existentes;
- Fix - Contém correcções concebidas.

Além disso, cada fix tem de pertencer obrigatoriamente a uma versão, podendo cada versão ter vários fixes.

A informação pertencente às notas de lançamento deve apresentar-se organizada e devidamente categorizada por versão/fix. Deve ser possível para o utilizador filtrar a informação por versão e fix.

Sempre que for realizada uma actualização ao produto (e, consequentemente, houver novidades nas Notas de Lançamento) deverá ser realizada uma notificação aos seus utilizadores. Esta deverá, ainda, oferecer a possibilidade de escolha entre consultar as mesmas ou continuar.

Descrição dos Casos de Uso

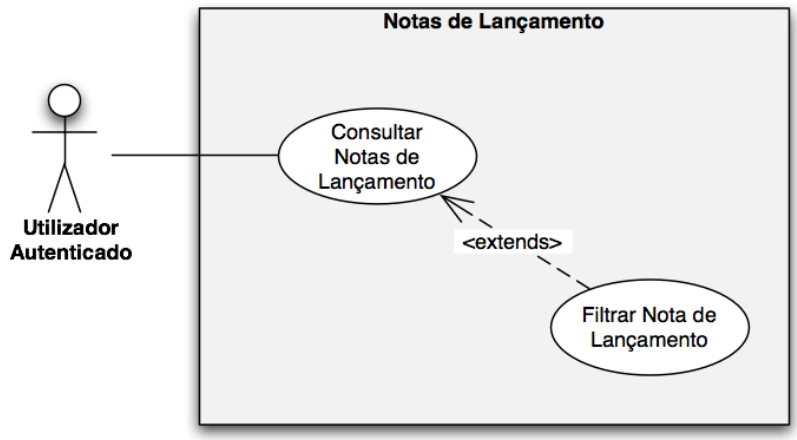

Figura 3.6: Diagrama de casos de uso das "Notas de Lançamento"

Consultar Notas de Lançamento

Possibilita que o utilizador consulte todas as notas de lançamento existentes no produto.

Filtrar Notas de Lançamento

Permite filtrar as notas de lançamento existentes por versão e/ou fix a que pertencem.

Diagrama de Estados

Este diagrama apresenta o fluxo normal quando existe uma nova nota de lançamento. Posteriormente, o utilizador poderá continuar a consultar as notas de lançamento acedendo directamente a "Apresenta Notas de Lançamento para Versão/Fix Seleccionado".

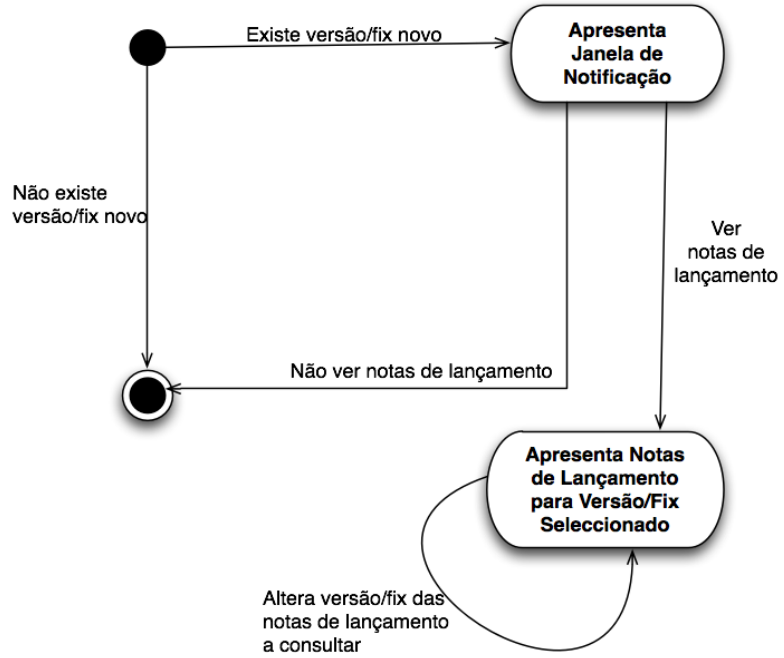

Figura 3.7: Diagrama de estado das "Notas de Lançamento"

3.1.3.3 Calculadoras

Esta funcionalidade está destinada a agregar diversos tipos de calculadoras que realizam cálculos estatísticos relacionados com a saúde e bem-estar.

Deve ser disponibilizada uma organização por categorias de calculadoras. Após seleccionada uma categoria, deverá surgir então uma listagem com todas as suas calculadoras. Cada uma delas disponibiliza um menu que permite o acesso à realização de um novo cálculo e à consulta do último resultado obtido.

Há uma necessidade forte para a existência de um elevado grau de escabilidade, uma vez que é previsto que o número de calculadoras vá aumentando gradualmente.

Inicialmente, acompanharão o lançamento desta funcionalidade as categorias gestão de peso, estilo de vida e saúde da mulher. Cada uma respectivamente com as

calculadoras: índice de massa corporal, calorias diárias recomendadas e data provável de parto.

Descrição dos Casos de Uso

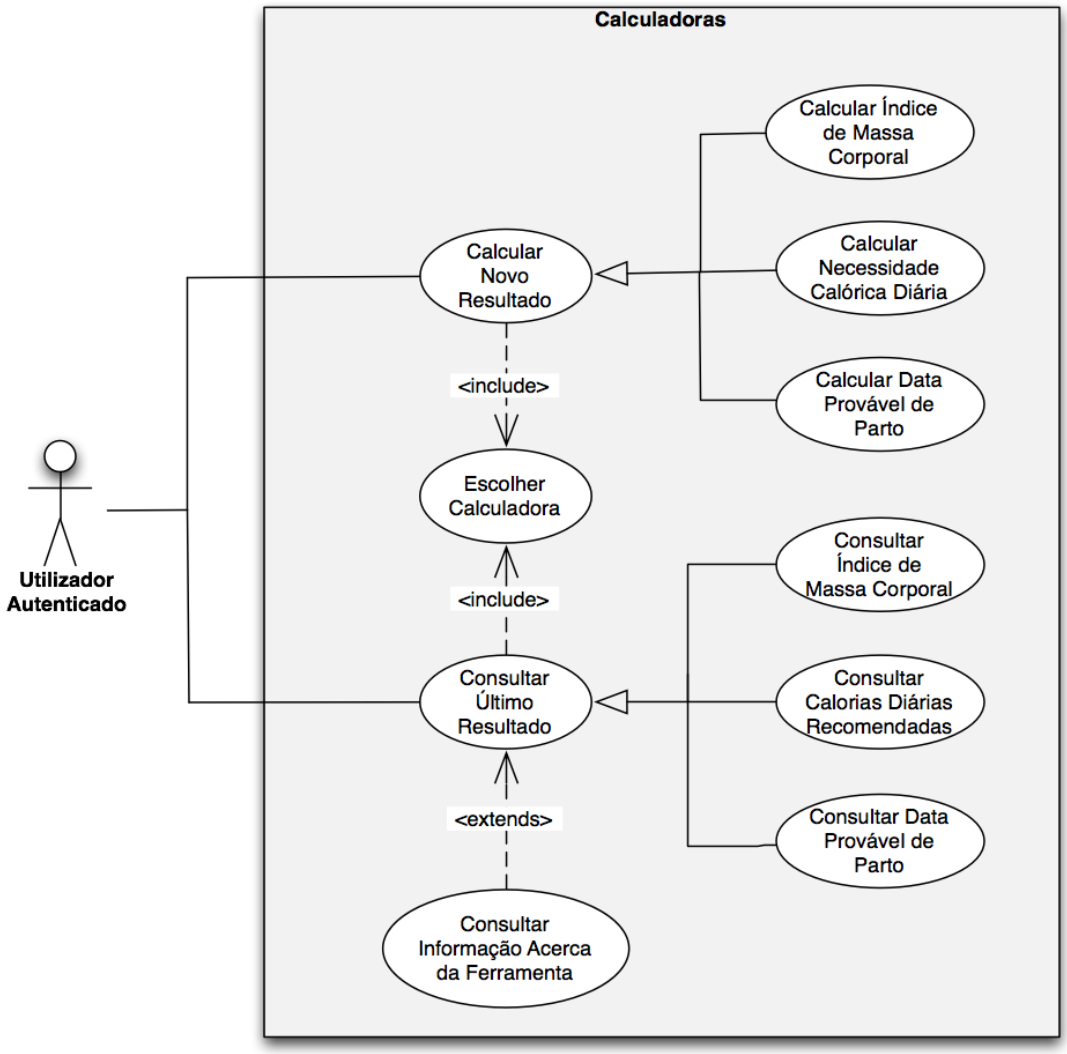

Figura 3.8: Diagrama de casos de uso das "Calculadoras"

Calcular Novo Resultado

Permite que o utilizador realize os cálculos na calculadora escolhida.

Consultar Informação Acerca de Ferramenta

Oferece a possibilidade de consultar informações sobre os cálculos realizados por uma calculadora.

Consultar Ultimo Resultado

Possibilita a consulta do último resultado obtido numa calculadora, desde que já exista um cálculo prévio na calculadora em questão.

Diagrama de Estados

No seguinte diagrama de estados é apresentada a navegação para esta funcionalidade. Como é possível observar, é possível aceder a qualquer momento a todos os menus, independentemente do menu correntemente seleccionado.

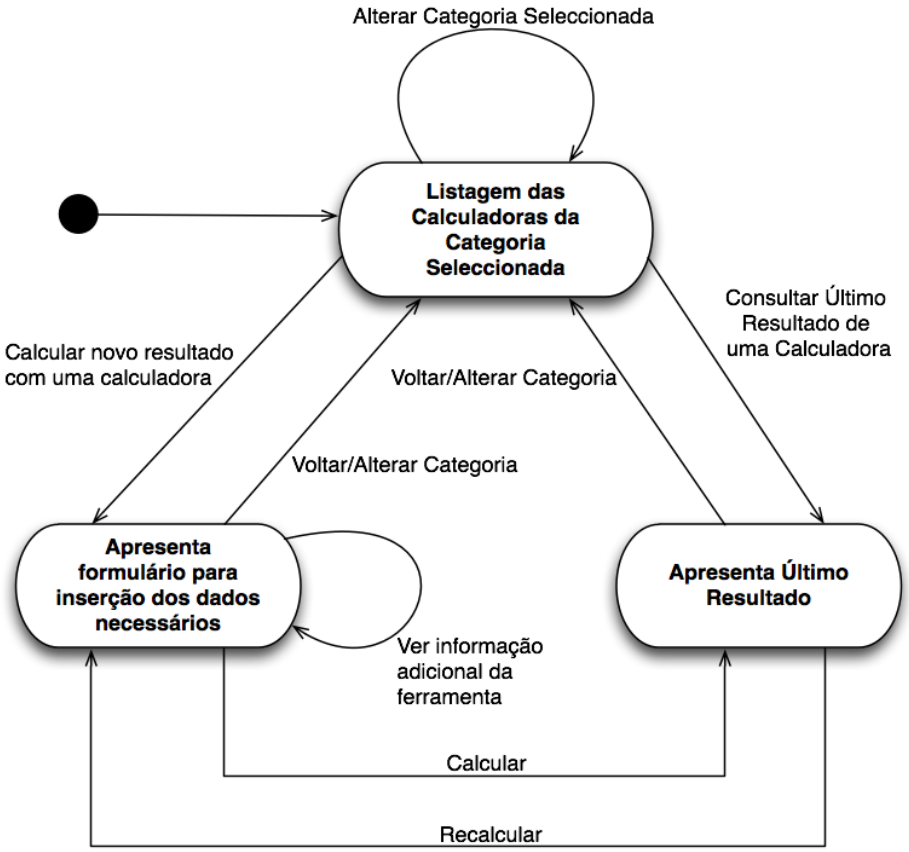

Figura 3.9: Diagrama de estado da funcionalidade "Calculadoras"

4 Desenvolvimento

Neste capítulo é descrita toda a arquitectura e implementação dos módulos funcionais desenvolvidos. São também apresentados aspectos fundamentais para se compreender como se processa o desenvolvimento em Flash 8, explicado como se extende e evolui a framework do ALERT PHR e, ainda, como e onde os módulos funcionais devem ser integrados no produto.

4.1 Conceitos Fundamentais para o desenvolvimento em Flash 8

Como a maioria das tecnologias de desenvolvimento, o Flash 8 Professional possui um conjunto de conceitos fundamentais que são necessários compreender para que se possa desenvolver com sucesso aplicações nesta tecnologia. De seguida, são apresentados os mais importantes para o desenvolvimento neste projecto.

Flash ActionScript 2.0

ActionScript trata-se de uma linguagem desenvolvida pela Macromedia para a plataforma Flash. Actualmente pertence à Adobe Systems Incorporated. A primeira versão foi lançada em Setembro de 2000, passados três anos, em Setembro de 2003, é actualizada para a versão 2.0 - versão utilizada no desenvolvimento deste projecto. Por fim, em Junho de 2006 é lançado o ActionScript 3.0 – que ainda hoje permanece como o estado da arte [FAS09].

O ActionScript possibilita uma programação eficiente de aplicações sobre a plataforma Flash.

Para usar Flash não é necessário recorrer a ActionScript, mas se a intenção for fornecer um grande dinamismo de dados e interacção com *back-ends*, o seu uso torna-se obrigatório.

O ActionScript 2.0 trata-se de uma linguagem de programação orientada a objectos (oriented programming language - OOP). Assim, oferece suporte para conceitos fundamentais de OOP, tais como:

- Classes;

- Herança;
- Interfaces.

As suas variáveis não são obrigatoriamente tipadas, no entanto, opcionalmente é possível restringi-las a um determinado tipo [ASG09].

Eventos

Os eventos consistem em sinais que são difundidos pelo Flash Player quando diversas acções orientadas ao utilizador ou ao sistema ocorrem. Estes podem ser tratados ou ignorados. Grande parte do desenvolvimento em Flash consiste na escrita de código que é apenas executado quando um evento específico é difundido por um determinado objecto. Esse código é denominado de "Event Handler" [ASG09].

Movie Clips

O símbolo Movie Clip é bastante poderoso e versátil. Possibilita que seja construída uma display list. Cada nó da mesma pode possuir conteúdo (raiz) ou conter outro nó. Ou seja, implementa o padrão de desenho de software Composite.

No Flash tudo se encontra dentro de um Movie Clip. Assim, no mínimo, existe um que se trata da raiz de todos os outros.

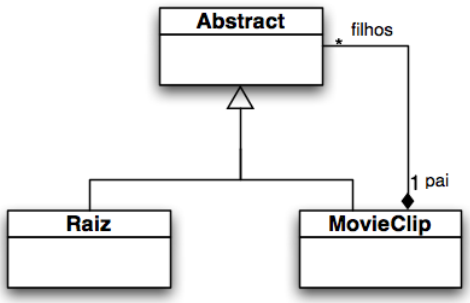

Figura 4.1: Padrão de desenho Composite aplicado ao MovieClip

Todos os Movie Clips têm associados a si uma linha temporal (timeline) que pode a cada instante controlar e actualizar o estado de cada um. No entanto, neste projecto não é necessária a sua utilização, geralmente é mais utilizado por programadores que apenas usam o Flash IDE sem recorrer à linguagem de ActionScript [DNAS09].

Layers

A layers são úteis para organizar conteúdos dentro de Movie Clips. Facilitam a organização, o que priveligia a manutenção do código e a criação de Movie Clips complexos. É, por isso, prática comum colocar em cada layer conteúdo relacionado. As layers tratam-se de profundidades de topo configuradas no IDE [ASG09].

Componentes

Os componentes tratam-se de módulos empacotados, reusáveis, capazes de disponibilizar uma ou mais funcionalidades. O seu comportamente e estado é

controlado por Action Script. Tratam-se de alicerces ("building blocks") que podem ser adicionados numa aplicação.

Os componentes desenvolvidos para este projecto pertencem a um tipo específico, são de user-interface (UIComponents). A título de exemplo, são exemplos deste tipo de componentes: radio buttons, dialog box, check boxes, dialog boxes, etc.

A arquitectura dos componentes é completamente extensível, permitindo que seja fácil para os programadores criarem os seus próprios componentes de modo a preencher completamente os seus requisitos específicos [DNAS09].

4.2 Arquitectura e Implementação

Seguidamente serão descritos todos os detalhes relevantes relativos à arquitectura e implementação das soluções que foram propostas e como estas foram integradas no produto.

4.2.1 Componentes

4.2.1.1 Desenvolvimento de UIComponents em Flash 8

Para desenvolver um *UIComponent* em Flash AS 2.0, a classe *UIComponent* deverá ser extendida. Esta não representa um componente visual, mas contém métodos, propriedades e eventos comuns a todos os componentes - o que permite partilhar o comportamente que têm em comum.

Existe um conjunto de métodos cujo o override pode ser bastante útil por parte do componente a ser desenvolvido. Assim:

- **public function init() : Void**
  Quando uma instância de um classe do *UIComponent* é criada, o Flash chama este método. É apenas chamado uma vez.

- **private function createChildren():Void**
  Os componentes implementam este método quando possuem a necessidade de criar subobjectos (tais como outros componentes) no próprio componente.

- **private function size():Void**
  Quando é feito redimensionamento de um componente em runtime através do método setSize(), este método é invocado e são-lhe passadas as propriedades de largura e altura. Deste modo, este método pode ser usado para reajustar o posicionamente do seu conteúdo. No mínimo, este método tem que chamar o método size da sua superclasse.

- **private function draw():Void**

Este método possui como finalidade a criação e modificação de elementos visuais do componente. Ou seja, neste método o componente redesenha-se para se actualizar de modo a corresponder às suas variáveis de estado.

4.2.1.2 Acerca desta Ferramenta

O "Acerca desta Ferramenta" consiste num componente bastante simples, ideal para iniciar o desenvolvimento na tecnologia.

Possui apenas dois estados: detalhes escondidos e ver detalhes. A parte gráfica do seu componente, nos seus seus dois estados possíveis, encontra-se ilustrada na figura seguinte.

Procedeu-se então à especificação do diagrama UML para este componente. Foi criada uma classe para o constituir, classe essa que extende a já existente *UIComponent*.

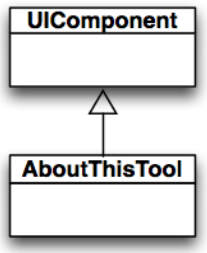

Figura 4.2: Diagrama de classes do componente "Acerca desta Ferramenta"

O componente é capaz de receber os seguintes parâmetros para a sua configuração:

- título;
- texto descritivo;
- endereço da localização da imagem de background.

Iniciou-se o desenvolvimento pelo desenho de toda a parte gráfica. Sempre que possível, todas as secções são desenhadas programaticamente em vez de recorrer ao carregamento de imagens. Isto permite que o tamanho final do componente seja menor e, consequentemente, demore menos tempo a carregar. Para isso recorreu-se a métodos estáticos da classe *PhrDrawUtils* disponibilizados pela framework do produto:

- **createSquare** – Método sem retorno. Apesar do seu nome, permite desenhar um rectângulo de comprimento e largura a especificar através da passagem de argumento. Tal como, o nível de transparência, grau de curvatura dos vértices e cores em gradiente.
- **drawGradientPolygon** – Desenha um polígono. Os argumentos que recebe são semelhantes aos do createSquare, no entanto em vez da a largura e comprimento, recebe um *array* com as coordenadas de cada vértice seu.

Para realizar o carregamento da imagem de background que se encontra no url especificado, utilizou-se o método *loadClip* da classe *MovieClipLoader* do Flash.

De seguida, foi necessário aplicar os estilos às fontes das caixas de texto utilizadas. Estes encontram-se definidos num ficheiro XML e podem ser aplicados através da utilização do método da classe *PhrUtil*:

- **configTextField** – permite aplicar estilos nas caixas de texto e configurar parâmetros relativos ao modo de impressão.

Após a conclusão do desenvolvimento da parte gráfica, instalou-se um handler para o evento de clique no *MovieClip* da secção de título. Caso o *MovieClip* com o texto descritivo se encontrar escondido, o hander mostra-o; caso já se encontre visível, o handler esconde-o.

Foi ainda instalado um handler para o evento de "*mouse over*" sobre a mesma secção. Este permite a alteração da cor de fundo do mesmo conforme o rato se encontrar sobre ou fora de si.

4.2.1.3 Título Dinâmico

O desenvolvimento deste componente foi realizado após o "Acerca desta Ferramenta". Trata-se, igualmente, de um componente bastante simples, ideal para continuar a familiarização com o Flash 8.

Este componente apresenta dois estados possíveis: subtítulo escondido e subtítulo visível.

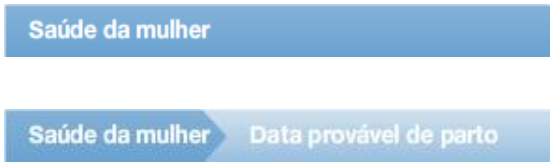

Figura 4.3: Dois estados possíveis para o componente "Título Dinâmico"

Para a implementação deste componente foi necessária a criação de uma nova classe que extende a *UIComponent*.

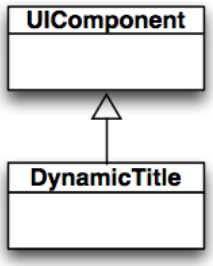

Figura 4.4: Diagrama de classes do componente "Título Dinâmico"

Os conteúdos do título e do subtítulo encontram-se em *MovieClips* diferentes que inicialmente se encontram em posições adjacentes. No entanto, a zona de visibilidade está limitada e apenas permite observar a secção do título. Para obter este resultado utilizou-se o método *setMask* da classe *MovieClip*.

Figura 4.5: Área de visibilidade do "Título Dinâmico"

Finalmente, para fazer surgir o subtítulo basta movimentar, na horizontal, o seu *MovieClip* para a área de visibilidade. Utilizando, para isso, a classe *Tween* do Flash. O efeito obtido encontra-se representado na figura seguinte.

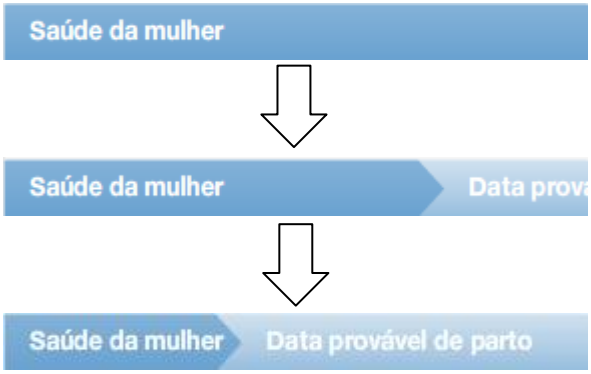

Figura 4.6: Movimento de aperecimento do subtítlo do componente "Titulo Dinâmico"

Quando o subtítlo se encontra vísivel, está activo um handler para o evento de clique sobre o componente. Esta acção origina com que o subtítulo se esconda.

4.2.1.4 Menu da Calculadora

A implementação do "Menu da Calculadora" veio finalizar o periodo de adaptação inicial à tecnologia e o desenvolvimento de componentes/funcionalidades simples.

Apresenta dois estados distintos. No mais complexo, faculta dois botões. Quando o utilizador realiza um clique sobre um deles, é lançado um evento que permite que as instâncias que se encontrem à escuta sejam notificadas dessa acção.

Figura 4.7: Exemplos dos dois estados possíveis do componente "Menu da Calculadora"

Os eventos são lançados através do recurso ao método *dispatchEvent* da classe *MovieClip* e os seus nomes são:

- *result;*
- *calculator.*

As classes que pretendam establecer um handler para estes eventos deverão faze-lo através da chamada do método *addEventListener* sobre a instância que possuem do componente. Deverão especificar como argumento o nome do evento a que se referem e o método que será o handler.

O estado inicial do componente é especificado pela passagem de um argumento na sua função de inicialização, assim como todo o conteúdo textual que este apresenta.

4.2.1.5 Calendário

Relativamente aos componentes anteriores, o desenvolvimento deste é bastante mais complexo, pois possibilita várias configurações distintas. Por defeito, permite ao utilizador seleccionar uma data e hora (que pode estar no formato am/pm ou 24h).

Figura 4.8: Componente do Calendário

No momento da sua inicialização é-lhe passado por argumento uma máscara no formato string. Esta pode conter dois caracteres distintos:

- **D** – Significa que é pretendido que o componente disponibilize a selecção de uma data;
- **H** – Permite seleccionar horas e minutos.

Realizando as combinações possíveis com os caracteres do argumento de entrada é possível definir que campos de selecção são disponibilizados e qual a sua ordem. Por exemplo, a string “DH” significa que o componente irá apresentar o campo de selecção da data, seguido do da hora ; por sua vez, a string “H” especifica que apenas irá ser apresentado o campo de selecção da hora.

O campo de selecção de hora é constituído por duas combo boxes diferentes: uma para permitir a selecção da hora do dia e outra a dos minutos com granularidade de cinco. Ambas são instâncias do componente *PhrComboBox* da framework do produto.

Caso o utilizador tenha definido nas suas preferências pessoais que prefere o formato da hora em am/pm, são adicionados dois radio buttons da classe pertencendo à framework do produto.

Figura 4.9: Componente do "Calendário" a permitir apenas a inserção de hora

Por sua vez, a selecção da data é feita através de um calendário desenvolvido para o efeito. Este surge, em forma de *pop-up*, quando o utilizador realiza um clique sobre a região assinalada na figura seguinte.

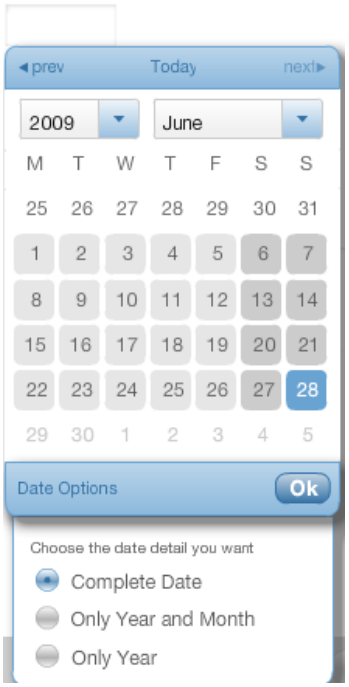

Figura 4.10: *Pop-up* do componente "Calendário" que permite a selecção de uma data

Na implementação deste componente foi criada a classe *PhrNewDate* que extende a *UIComponent*. Possui um *pop-up* de calendário, *DatePopUp*, que possibilita a escolha de uma data. As classes *CalendarDate* e *Calendar* oferecem-lhe suporte para a concretização da sua tarefa.

Foi também implementada uma última classe, *GregorianCalendar*, que algomera um conjunto de métodos para operações sobre datas em calendários gregorianos.

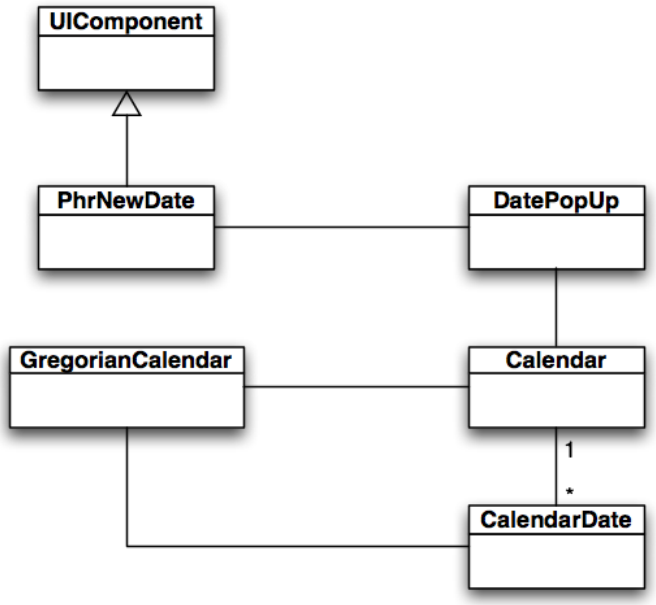

Figura 4.11: Diagrama de classes do componente "Calendário"

GregorianCalendar

Esta classe consiste numa API que disponibiliza diversas operações sobre calendários gregorianos, nomeadamente:

- **Obter dia da semana para uma data especificada:**

  Como é natural, a uma determinada data (dia, mês e ano) corresponde um dia da semana (por exemplo, Segunda-Feira). Para realizar este cálculo, este método implementa o algoritmo de *Doomsday*, cujo o pseudo-código é apresentado de seguida.

Tabela dos Meses	
Janeiro	0 (6 em anos bissextos
Fevereiro	3 (2 em anos bissextos)
Março	3
Abril	6
Maio	1
Junho	4
Julho	6
Agosto	2
Setembro	5
Outubro	0
Novembro	3
Dezembro	5

Tabela dos dias	
Domingo	0
Segunda	1
Terça	2
Quarta	3
Quinta	4
Sexta	5
Sábado	6

Figura 4.12: Informação prévia necessária para correr o algoritmo de *Doomsday*

1) Calcular $c = 2(3 - (\text{sec} \,\%\, 4))$, onde sec são os primeiros dois digitos do ano;
2) Seleccionar os últimos dois dígitos do ano;
3) Dividir o resultado obtido em 2) por 4 e manter apenas o quociente;
4) Obter valor associado ao mês na Tabela dos meses;
5) Adicionar o dia do mês aos resultados obtidos de 1) a 4)
6) Dividir o resultado de 5) por 7 e manter o resto;
7) Procurar na Tabela dos dias, qual o dia com o valor associado encontrado em 6) [DDA09].

- **Validar uma data:**

  Garante que uma data especificada por argumento é válida. Por exemplo, 12 de Janeiro de 1999 é uma data válida, porém 31 de Fevereiro de 2009 não o é;

- **Obter número de dias de um determinado mês**:

  Devolve o número de dias que um determinado mês teve num determinado ano. Naturalmente, só é necessário especificar o ano se o mês a analisar for Fevereiro.

- **Verificar se um ano é ou não bissexto.**

CalendarDate

Consiste numa estrutura de dados para o armazenamento de datas. Fornece métodos para a sua manipulação (alterar dia, mês ou ano) e comparação de grandeza, ou seja, verifica se uma data é posterior ou anterior a outra.

Para garantir que as datas que lhe são especificadas são válidas, recorre à API da classe GregorianCalendar.

Calendar

Esta classe é responsável pelo desenho da interface gráfica que permite ao utilizador seleccionar uma data.

Figura 4.13: Zona desenhada pela classe Calendar do componente "Calendário"

Recapitulando, a selecção de uma data poderá estar limitada por uma data passada e/ou futura. Poderá também possuir diferentes níveis de detalhe:

- Dia, mês e ano;
- Mês e ano;
- Apenas ano.

As datas limites de passado/futuro e a data correntemente seleccionada são guardadas em váriaveis da classe que consistem em instâncias de *DateCalendar*.

A área de selecção do dia, é constituída por uma grelha de 6 linhas por 7 colunas, onde irá constar em cada posição um dia específico. Para realizar o seu desenho, foi desenvolvido o seguinte algoritmo:

1) Obter dia da semana para o primeiro dia do mês correntemente seleccinado;
2) Iniciar o preenchimento na primeira linha no dia da semana obtido em 1) até atingir o final do mês;
3) Completar a grelha com os dias do mês seguinte;
4) Obter número de dias do mês anterior;
5) Iniciar o preenchimento, de modo descrescente, no dia prévio ao obtido em 1) até atingir o inicio da grelha.

À medida que este algoritmo é executado, é identificado o devido estado para cada dia e só depois é feito o respectivo desenho. Os estados possíveis, ordenados por prioridade, são os seguintes:

- **Inactivo** – Não é possível seleccionar um dia que se encontre neste estado. Sucede quando este se encontra fora do intervalo permitido pelas datas limite ou quando o detalhe da data apenas inlcui mês e ano;
- **Dia do mês anterior/próximo** – Quando o dia a desenhar não pertence ao mês escolhido. O clique por parte do utilizador sobre este dia, origina não só a sua selecção, como também a alteração do mês a ser apresentado no calendário;
- **Dia de Segunda-Feira a Sexta-Feira** – Dia do mês seleccionado, com a particularidade de se tratar de um dia útil;
- **Dia de Fim de Semana** - Dia de fim de semana do mês seleccionado.

DatePopUp

Na classe *DatePopUp*, é construída toda a interface que permite ao utilizador especificar uma data e escolher o seu detalhe.

Este é colocado numa layer que se situa sobreposta ao restante conteúdo da aplicação e que é partilhada por todos os *pop-ups* existentes. Isto garante que todos surjam no topo, sem nada que os obstrua.

Por questões de eficiência, esta classe implementa o padrão de desenho Singleton. Isto permite que todos os componentes da data existentes na aplicação partilhem o mesmo DatePopUp, em vez de cada uma criar e usar o seu. Para isso, esta classe dispara um evento quando uma data é seleccionada pelo o utilizador, o que origina com que o componente que tenha subscrito esse evento seja notificado e receba os dados inseridos.

4.2.2 Funcionalidades

O ALERT® PHR é consituído por três zonas principais:

- Header;
- Screen;
- Viewer.

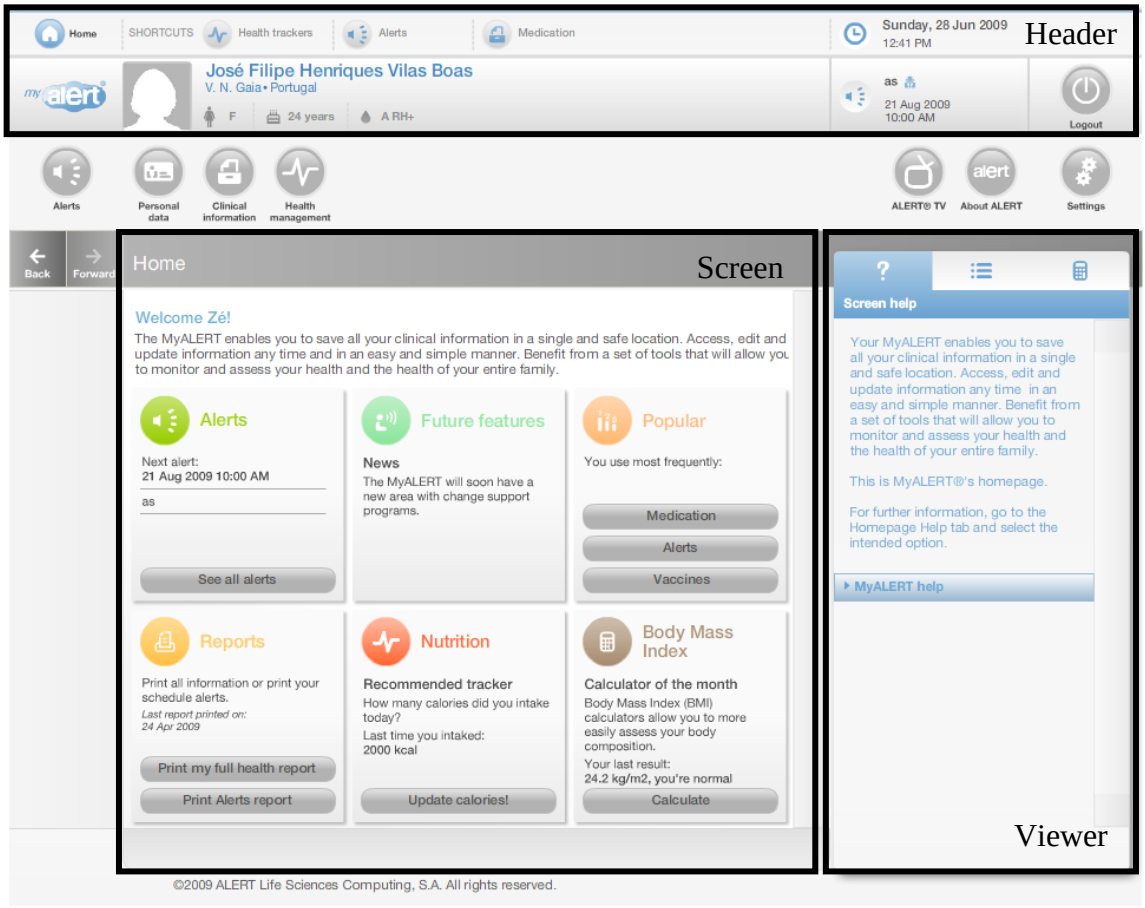

Figura 4.14: Zonas principais constituintes do ALERT® PHR

Desenvolver módulos funcionais para estas diferentes zonas, requer o seguimento e respeito da estrutura específica de cada uma. Só assim, é possível que as suas funcionalidades sejam extendidas com sucesso.

Para este projecto, foram desenvolvidas novas funcionalidades para a zona de Screen e Viewer.

4.2.2.1 Viewer

O Viewer é uma zona pensada e desenvolvida para proporcionar um acesso rápido, simples e intuitivo às várias funcionalidades que disponibiliza. Foi desenvolvido tendo como objectivo crucial a escalabilidade.

Implementa o padrão de desenho *Abstract Factory*.

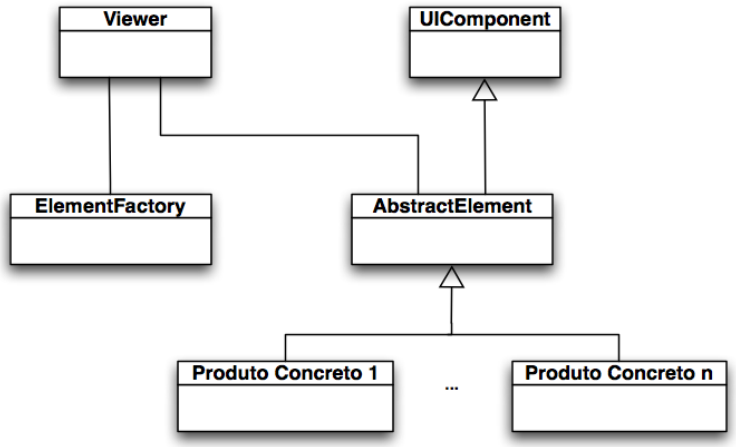

Figura 4.15: Diagrama de classes do Viewer

As funcionalidades pertencentes ao Viewer encontram-se organizadas por tabs. Inicialmente estas são obtidas da base de dados e de seguida desenhadas. Para cada uma é extraído o seu conteúdo através de uma fábrica, classe *ElementFactory*, que recebe como argumento o identificador de cada tab e retorna uma instância da classe com o seu conteúdo. Todas essas classes retornadas extendem a *AbstractElement*.

Isto permite que toda a configuração fique na base de dados, o que é excelente em termos de escalabilidade.

Assim, para iniciar o desenvolvimento de uma nova funcionalidade para o Viewer é necessário adicionar previamente a informação relativa à tab respectiva na base de dados.

4.2.2.1.1 Ajuda Contextualizada

O conteúdo de ajuda desta funcionalidade é automáticamente adaptado à actual localização do utilizador no produto.

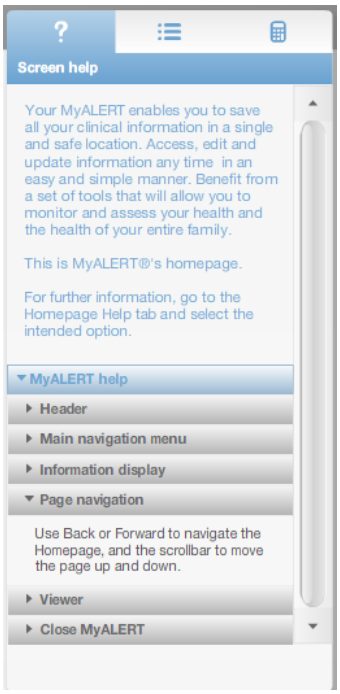

Figura 4.16: Exemplo da funcionalidade "Ajuda Contextualizada"

Sempre que o utilizador navega entre os diferentes menus da aplicação, é realizado um novo pedido à base de dados para obter toda a ajuda de determinado menu. Esta informação possui uma descrição genérica e detalhes pormenorizados, que se encontram organizados categoricamente de forma hierárquica. Deste modo, tendo em conta o formato do tipo de dados, optou-se por carregar para uma estrutura de dados em árvore toda a informação, pois é a que a melhor simula a sua estrutra inicial.

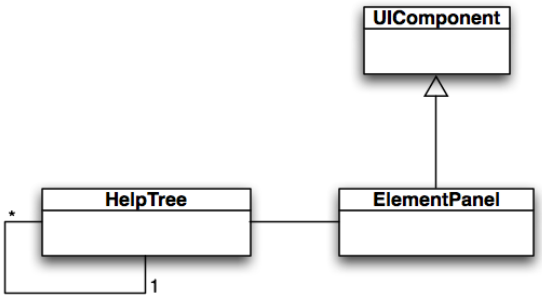

Figura 4.17: Diagrama de classe da funcionalidade "Ajuda Contextualizada"

Uma vez que a informação de ajuda se encontra organizada por categorias, estas podem-se encontrar abertas para consulta – neste caso o seu conteúdo surge imediatamente abaixo de si; ou fechadas – o seu conteúdo não é mostrado. A alternação deste estado é responsabilidade do utilizador, que para o fazer basta clicar nas categorias.

A representação gráfica desta funcionalidade é obtida através do seguinte algoritmo recursivo:

- Imprime descrição genérica
- Para a raiz da árvore chama:
    - Imprime nó da árvore:
        - Se tiver filhos,
            - Desenha secção com o título da categoria, na posição seguinte;
            - Se a categoria estiver aberta para consulta,
                - Para cada filho chama “Imprime nó da árvore”;
        - Se não tiver filhos (for folha),
            - Desenha secção com a informação descritiva que lhe pertencente;

4.2.2.1.2 Calculadoras

Esta funcionalidade disponibiliza um conjunto de calculadoras. A par do desenvolvimento desta funcionalidade, foram desenvolvidas três calculadoras iniciais para a acompanhar:

- **Gestão do peso** – Calculadora de BMI;
- **Estilo de Vida** – Calculadora de consumo diário de calorias recomendado;
- **Saúde da Mulher** – Calculadora de data prevista de parto.

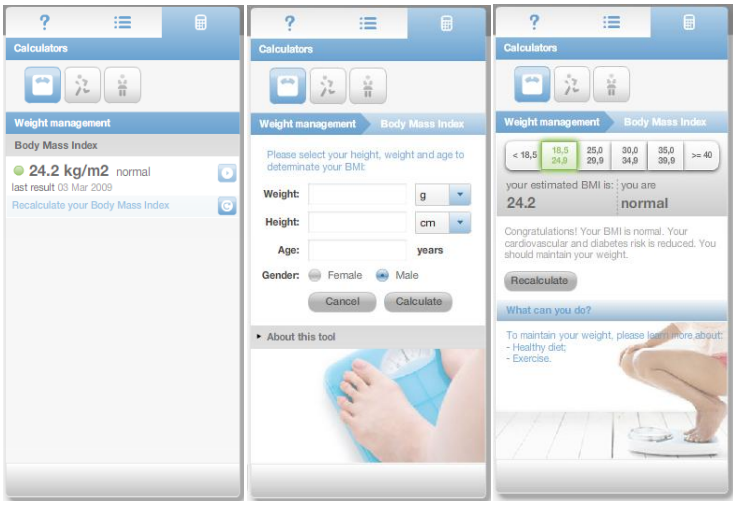

Figura 4.18: Exemplo da funcionalidade "Calculadoras"

Na implementação desta funcionalidade, foram utilizados quatros componentes desenvolvidos neste projecto e descritos anteriormente neste capítulo:

- Acerca desta Ferramenta;
- Título Dinâmico;
- Menu de Calculadora;
- Calendário.

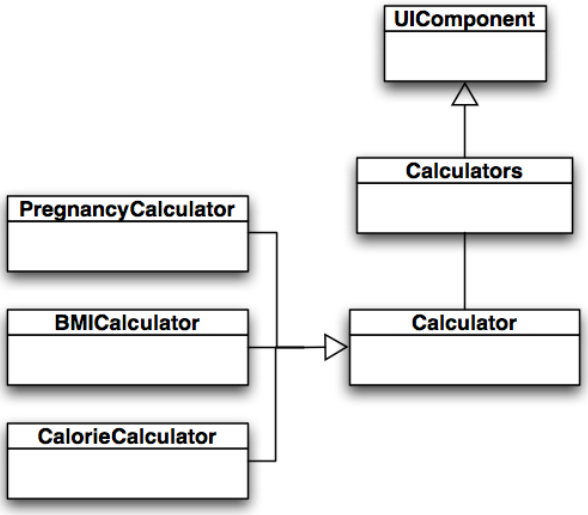

Figura 4.19: Diagrama de classes da funcionalidade "Calculadoras"

Para começar, a classe *Calculators* obtém informação de todas as categorias e as suas respectivas calculadoras da base de dados. De seguida, utiliza as categorias obtidas para criar a barra de navegação entre elas.

Figura 4.20: Zona de selecção da categoria da calculadora a usar

De seguida é adicionado o componente “Título Dinâmico” que parte para a criação da zona onde surgem as diversas calculadoras que podem ser escolhidas. Após a selecção de uma delas, o conteúdo dessa zona é alterado e passa a conter a calculadora escolhida.

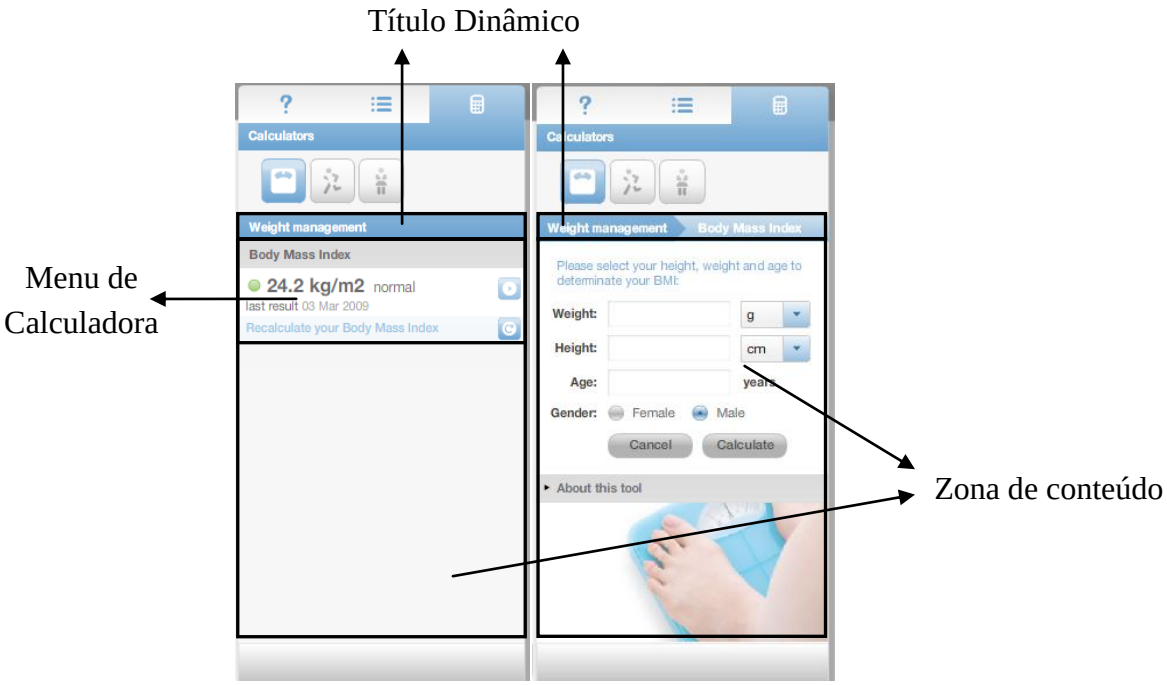

Figura 4.21: Legendas das diferentes secções constituintes da calculadora

Por fim é adicionado o componente "Menu da Calculadora" à zona de conteúdo para cada uma das calculadoras e são instalados dois handlers para os eventos lançados pelos cliques em cada botão desse componente. Um deles irá permitir fazer um novo cálculo na calculadora respectiva e o outro consultar detalhes do último resultado obtido.

As calculadoras implementadas extendem a classe *Calculator*. Devem possuir um Movie Clip que contenha a zona de formulário da calculadora e outro com o zona de apresentação de resultados. Devem, ainda, lançar eventos após a finalização do seu carregamento inicial após serem invocadadas e quando o utilizador realiza acções de calcular resultado, cancelar cálculo e retroceder. A classe *Calculators* implementa os handlers de resposta a esses eventos e com base neles gere o fluxo de navegação, adequando o devido ecrã à zona de conteúdo a cada momento.

4.2.2.2 Screen

Quando a aplicação do ALERT® PHR é inicialmente carregada, obtém da base de dados todos os botões a que o utilizador que se autenticou tem acesso, bem como os respectivos deepnavs.

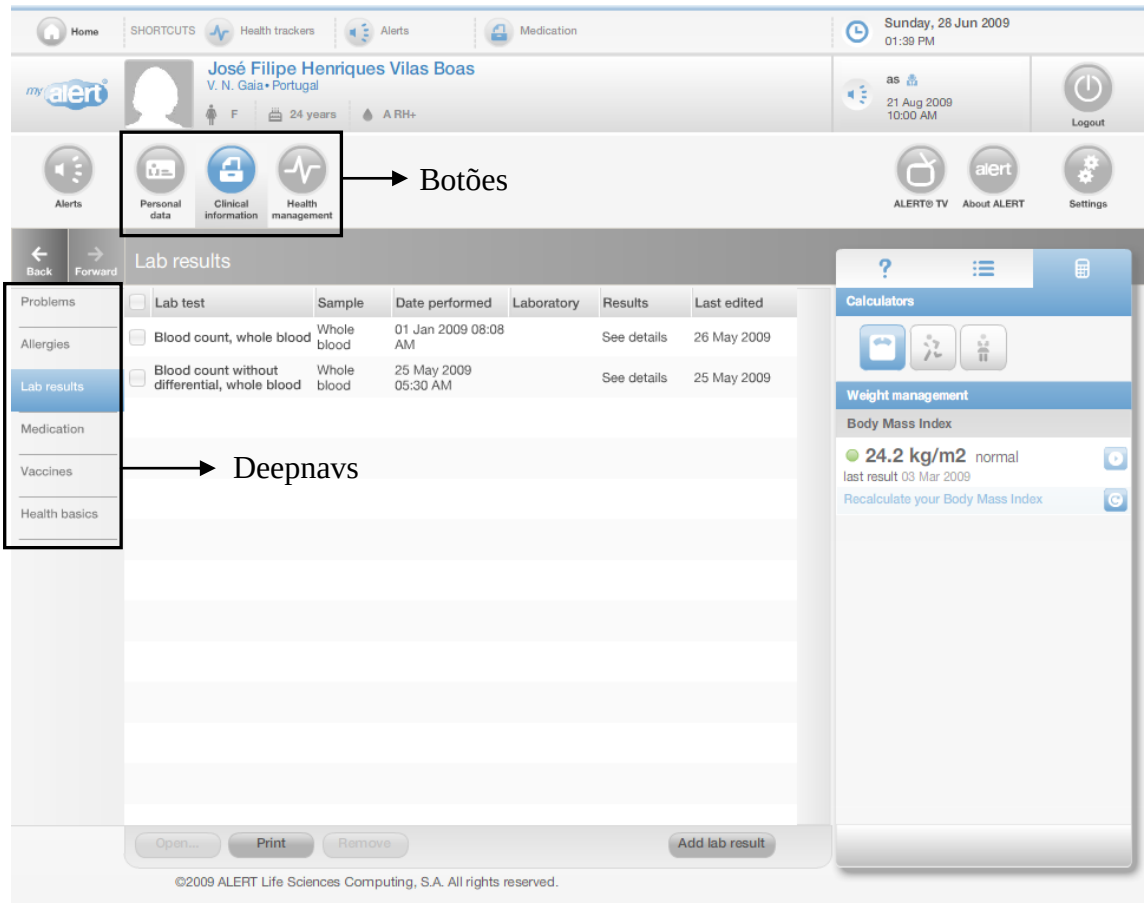

Figura 4.22: Botões e Deepnavs do ALERT® PHR

Deste modo, para desenvolver um novo Screen é necessário que se insira na base de dados um deepnav com um botão associado, juntamente com a informação do nome do ficheiro binário (SWF) que implementa o Screen. Isto permite que, quando um deepnav é seleccionado por um utilizador, se possa proceder à chamada do Screen respectivo.

Para implementar com sucesso um Screen, deve extender-se a classe Screen e implementar a interface IScreen, que define um conjunto de métodos que devem ser implementados por todos os screens.

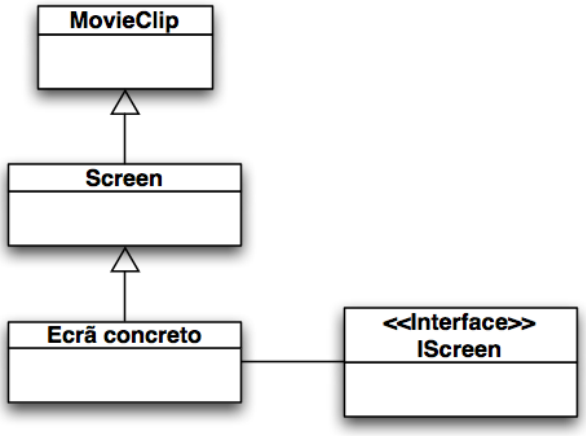

Figura 4.23: Diagrama de classes do "Screen" do ALERT® PHR

Os métodos a implementar definidos na interface possuem as seguintes finalidades:

- Realizar a iniciação do Screen;
- Inicialização de variáveis;
- Fecho do screen (sucede quando o utilizador navega para outra funcionalidade);
- Verificar se os dados inseridos no Screen se encontram guardados.

Nos capitulos seguintes são descritas as funcionaldades desenvolvidas no contexto da área de conteúdos (screen).

4.2.2.2.1 Notas de Lançamento

Esta funcionalidade é constituída por duas partes distintas:

- A primeira consiste num *pop-up* de aviso ao utilizador que o notifica do lançamento de uma nova versão do ALERT® PHR;
- A segunda, trata-se da consulta de todas as alterações/correcções realizadas ao produto desde a sua primeira versão.

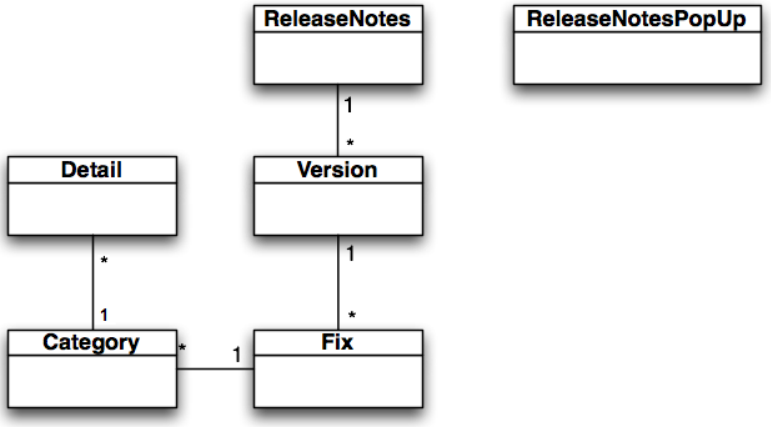

Figura 4.24: Diagrama de classes da funcionalidade "Notas de Lançamento"

Iniciou-se o desenvolvimento pela implementação do *pop-up* de aviso. Este é construido sobre uma *layer* definida incialmente onde são carregados todos os *pop-ups* de forma a não haver conflictos de profundidades. A implementação do *pop-up* é feita pela classe *ReleaseNotesPopUp*.

Quando o produto arranca, este verifica na base de dados se existe alguma nova versão do produto sobre a qual o utilizador não tenha sido ainda notificado. Se isto se verificar, surge este *pop-up* de aviso que possibilita as acções de o utilizador continuar o seu percurso ou de consultar as notas de lançamento.

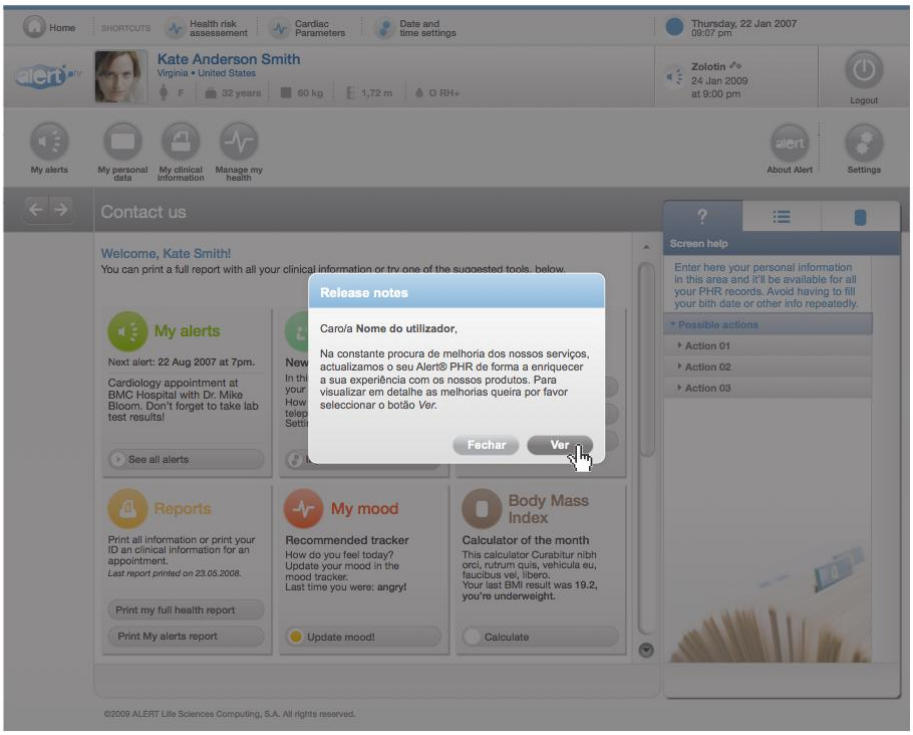

Figura 4.25: *Pop-up* de notificação de lançamento de nova versão/fix do ALERT® PHR

Finalizado o desenvolvimento do *pop-up*, iniciou-se o desenvolvimento do ecrã para a consulta das notas de lançamento. De início são transferidas da base de dados as notas de lançamento para todas as versões e fixs e, de seguida, carregadas para a sua estrutura de dados (ver diagrama de classes uml). Como é possível verificar, uma versão possui diversos fixs, onde a informação que cada um contém está organizada por várias categorias que lhe pertencem. Estas, por sua vez, possuem vários detalhes que englobam vários items explicativos.

Para possibilitar a escolha de um par versão/fix ao utilizador, são usadas duas combo box da framework do produto. Uma possibilita a escolha de versão e a outra a escolha dos fixs dessa mesma versão.

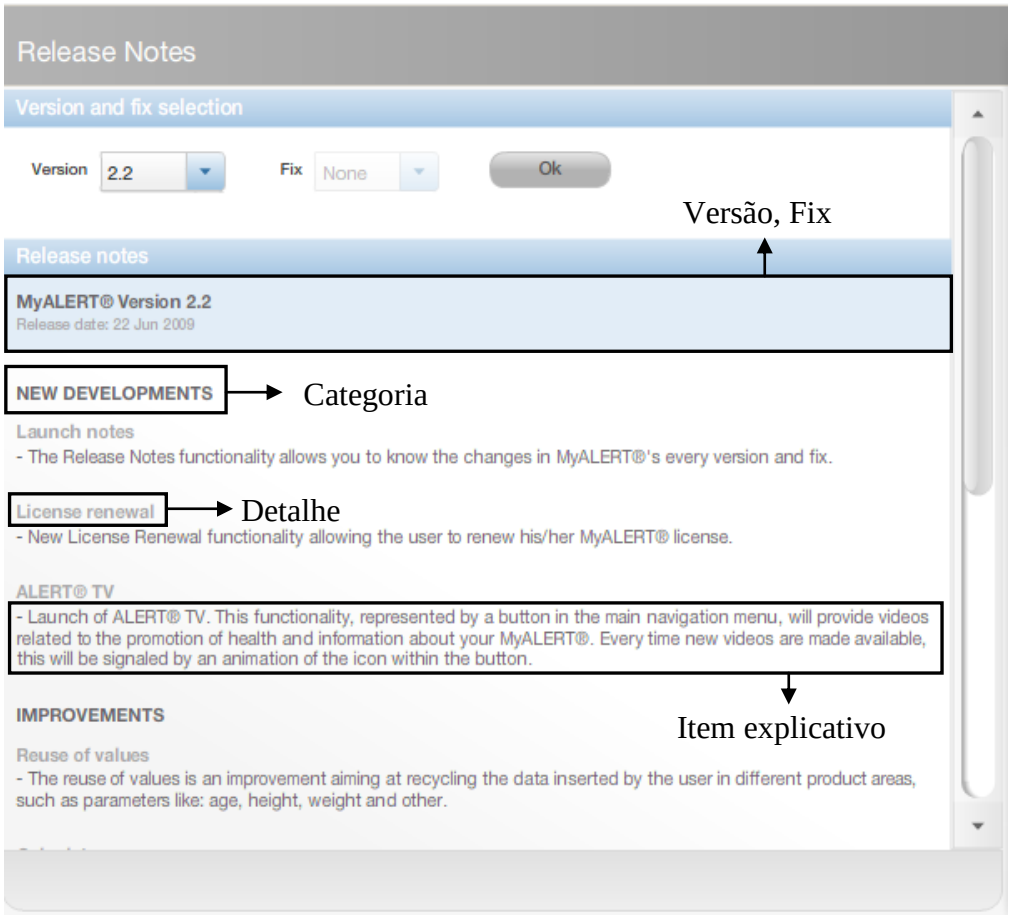

Figura 4.26: Legenda das diferentes secções que constituem a funcionalidade "Notas de Lançamento"

Após seleccionada uma versão e fix, é usado o seguinte algoritmo para proceder ao desenho estruturado e organizado do seu conteúdo respectivo:

- Para a versão e fix seleccionados:
  - Desenha secção de título principal
  - Para todas as categorias do fix:
    - Desenha título de categoria
    - Para todos os detalhes da categoria:
      - Desenha texto do detalhe
      - Para todos os items do detalhe:
        - Desenha o item explicativo.

5 Conclusões e Perspectivas Futuras

O desenvolvimento deste projecto revelou-se um desafio bastante motivador. Isto porque me permitiu aplicar diversas áreas de conhecimento adquiridas ao longo destes anos em que frequentei o mestrado. Além disso, integrei uma equipa de desenvolvimento, o que permite trocar experiências e opiniões para, em conjunto, procurar atingir um produto cada vez melhor.

Todas as funcionalidades propostas foram implementadas com sucesso no tempo planeado e encontram-se já disponíveis ao público em geral. O facto de terem sido desenvolvidas em Flash 8 representou uma dificuldade acrescida. É certo que Flash é uma tecnologia bastante maleável, que permite uma enorme configurabilidade dos aspectos gráficos, o que favorece o factor exclusividade de um produto, mas o desesenvolvimento torna-se demasiado moroso. Possui uma biblioteca muito limitada em estruturas de dados e é bastante tolerante a erros. Por vezes isto consiste numa mais valia, no entanto pode aumentar a duração do tempo de *debugging* e tornar-se num verdadeiro problema para o programador.

Actualmente, estão já disponíveis no mercado versões mais recentes do Flash, mas o principal problema persiste: o desenvolvimento é demasiado moroso. Nos dias de hoje não se trata da melhor opção para proceder ao desenvolvimento de RIAs, existem disponíveis tecnologias como Flex (também da Adobe) ou Microsoft Silverleight, orientadas especialmente para o efeito.

A equipa do ALERT® PHR pretende constantemente continuar a melhorar o produto de modo a obter diferenciação da concorrência e proporcionar uma melhor experiência aos seus utilizadores. Assim, o produto encontrar-se-á sempre em constante mutação, quer pela elaboração de funcionalidades novas, quer pela evolução das já existentes.

6 Bibliografia

[AOL09]. **ALERT-ONLINE.** ALERT-ONLINE, 2009. http://www.alert-online.com/.

[CNN09]. **CNN.** Obama's big idea: Digital health records, Janeiro de 2009. http://money.cnn.com/2009/01/12/technology/stimulus_health_care/.

[MHV09]. **HealthVault.** Microsoft HealthVault, 2009. http://www.healthvault.com/.

[GH09]. **Google Health.** Google Health, 2009. https://www.google.com/health.

[ADB09]. **Adobe.** Adobe, 2009. http://www.adobe.com/.

[AFL09]. **Wikipedia.** Adobe Flash, 2009. http://en.wikipedia.org/wiki/Adobe_Flash.

[LDCS09]. **LiveDocs.** Flash 8 LiveDocs, 2008. http://livedocs.adobe.com/flash/8/.

[WATC09]. **Wikipedia.** Apache Tomcat, 2009.
http://en.wikipedia.org/wiki/Apache_Tomcat.

[ATC09]. **Apache Tomcat.** Apache Tomcat, 2009. http://tomcat.apache.org/.

[ODB09]. **Wikipedia.** Oracle Database, 2009.
http://en.wikipedia.org/wiki/Oracle_Database

[ORC09]. **Oracle.** Oracle, 2009. http://www.oracle.com/index.html.

[ORP09]. **Oracle.** Oracle Database sets new performance records, 2009.
http://www.oracle.com/solutions/performance_scalability/database-se1-tpc.html.

[ECLPS09]. **Eclipse.** Eclipse.org, 2009. http://www.eclipse.org/.

[FDT09]. **Powerflasher Solutions.** Powerflasher FDT, 2009. http://fdt.powerflasher.com/.

[SBCPS09]. **Tigris.** Subclipse, 2009. http://subclipse.tigris.org/.

[XRAY09]. **OsFlash.** Xray (The AdminTool), 2009. http://osflash.org/xray.

[SCP09]. **KevinLangdom.** Service Capture, 2009.
http://www.kevinlangdon.com/serviceCapture/.

[JIRA09]. **Atlassian.** Jira Software, 2009. http://www.atlassian.com/software/jira/.

[FAS09]. **Wikipedia.** Flash ActionScript, 2009. http://en.wikipedia.org/wiki/ActionScript.

[ASG09]. **Adobe.** ActionScript Guide.
http://www.adobe.com/devnet/flash/articles/actionscript_guide.html.

[DNAS09]. **Adobe.** DevNet - ActionScript, 2009.
http://www.adobe.com/devnet/flash/actionscript.html.

[DDA09]. **Rudy.** Doomsday Algorythm, 2009. http://rudy.ca/doomsday.html.