

Faculdade de Engenharia da Universidade do Porto
Mestrado Integrado em Engenharia Informática e Computação



**Implementação da Intranet do Instituto da Construção e do
Imobiliário na
Companhia Portuguesa de Computadores
Informática e Sistemas, S.A.**

Relatório do Projecto Curricular do MIEIC 2007/2008

Ariel Dinis Meira Pestana

Orientador na FEUP: Prof. Ana Paula Rocha

Orientador na CPCis: Dr. Silvino Glória

Abril de 2008

"Em tudo na vida a perfeição é finalmente atingida, não quando nada mais existe para acrescentar, mas quando não há mais nada para retirar."

Saint-Exupéry

Resumo

O presente relatório pretende descrever a experiência de implementação da Intranet do Instituto da Construção e do Imobiliário (InCI). O InCI decidiu que era altura de criar uma nova plataforma de serviços, em substituição do sistema existente, procurando agilizar processos e transacções de negócio, permitindo ter uma visão alargada dos serviços prestados.

O projecto do aluno surge inserido na criação desta nova plataforma e tem como objectivo o desenvolvimento de funcionalidades para o portal de intranet, utilizando uma arquitectura orientada a serviços.

O projecto inclui uma prova de conceito sobre a plataforma de Windows *Workflow* Foundation; a construção do processo de Marcação de Férias gerido pela Direcção Administrativa, Financeira e de Recursos Humanos do InCI; funcionalidades de verificação de requisitos mínimos que é transversal a vários processos do Departamento de Qualificação; criação da gestão documental do instituto; geração automática de documentos assinados digitalmente com base em *templates*; e módulo de alertas.

Os objectivos propostos no planeamento do trabalho foram alcançados com sucesso, todas as funcionalidades foram desenvolvidas correctamente.

Acrónimos e Abreviaturas:

AD – Active Directory

BDC – Business Data Catalog

BI – Business Intelligence

CPCis – Companhia Portuguesa de Computadores Informática e Sistemas, S.A.

DAFRH – Direcção Administrativa, Financeira e de Recursos Humanos

DQ – Departamento de Qualificação

e.g. – do latim exempli gratia, significa – por exemplo

ECM – Enterprise Content Management

EDE – External Data Exchange

GUID – Globally Unique Identifier

HTTP – Hypertext Transfer Protocol

i.e. – do latim id est, significa – isto é

IIS – Internet Information Services

InCI – Instituto da Construção e do Imobiliário

MOSS – Microsoft Office Sharepoint Server

MWSS – Manual *Workflow* SchedulerService

PDC – Prova de Conceito

RST – Request Security Token

SI – Sistemas de Informação

SOA – Service Oriented Architecture

SOAP – Simple Object Access Protocol

SQL – Structured Query Language

SSO – Single Sign-On

STS – Security Token Service

VS – Visual Studio

WFF – Microsoft Windows *Workflow* Foundation

WI – *Workflow* Input

WH – *Workflow* Helper

WSDL – Web Services Description Language

WSS – Windows SharePoint Services

XML – eXtensible Markup Language

XSD – XML Schema Definition

Índice de Conteúdos

1	Introdução	1
1.1	O Projecto de Implementação da Intranet do InCI	1
1.2	Organização e Temas Abordados no Presente Relatório	2
2	Análise do problema.....	3
2.1	Projecto InCI	3
2.1.1	Instituto da Construção e do Imobiliário	3
2.1.2	Projecto a Desenvolver	3
2.2	Processo de Desenvolvimento de Software.....	5
2.2.1	Planeamento	5
2.2.2	Equipa de desenvolvimento	6
2.2.3	Metodologia.....	6
3	Revisão Tecnológica.....	8
3.1	Arquitectura Orientada a Serviços	8
3.1.1	Introdução	8
3.1.2	Benefícios essenciais da SOA	9
3.1.3	Propriedades de um Serviço	10
3.1.4	Princípios dos Serviços.....	12
3.1.5	Tipificação de Serviços	13
3.1.6	Peças constituintes da SOA.....	13
3.2	Funcionamento da Arquitectura Orientada a Serviços.....	15
3.2.1	Fases da Implementação de um Serviço	15
3.2.2	Comunicação Inter serviços.....	17
3.2.3	Gestão de Excepções	17
3.3	Plataforma tecnológica.....	18
3.3.1	Produtos e Tecnologias da Solução.....	18
3.4	Níveis de Serviço da Solução	22
3.5	Windows Workflow Foundation	22
4	Desenvolvimento do Projecto	26
4.1	Arquitectura lógica do projecto	26
4.2	ASP.NET e Microsoft Office SharePoint Server	27
4.3	Prova de conceito no Windows Workflow Foundation	28
4.3.1	Workflow Helper	29
4.3.2	Workflow Input	29
4.3.3	Serviço Workflow	30
4.3.4	Interface External Data Exchange.....	30
4.3.5	Serviço que implementa interface de EDE.....	31
4.3.6	Desenho do Workflow	31
4.3.7	Call Workflow Activity	32
4.3.8	Aplicação em caso real	32
4.4	Processo de Marcação de Férias do DAFRH	34
4.5	Processo de Concessão do Departamento de Qualificação	37
4.6	Gestão Documental	38
4.7	Geração automática de documentos	42
4.8	Alertas	44

5	Conclusões e perspectivas de trabalho futuro	46
6	Bibliografia.....	49
ANEXO A:	Detalhe do planeamento do projecto.....	50
ANEXO B:	Atributos de documentos	51

Índice de Figuras

Figura 1 - Planeamento das actividades do projecto	6
Figura 2 – Modelo de desenvolvimento de <i>Staged Delivery</i>	7
Figura 3 - Arquitectura Orientada a Serviços (SOA)	8
Figura 4 – Modelo de <i>Pipeline</i> do <i>Service Bus</i> [2]	14
Figura 5 - Registo e credenciação de um serviço	15
Figura 6 - Consumo de um <i>service provider</i>	16
Figura 7 - Modelo de consumo de serviço de negócio	17
Figura 8 - Produtos e Tecnologias da Solução	18
Figura 9 - Componentes do Microsoft Office Sharepoint 2007 [2]	20
Figura 10 - Exemplo de <i>workflow</i> sequencial	24
Figura 11 - Exemplo de <i>workflow</i> de máquina de estado	25
Figura 12 - Arquitectura lógica do projecto	26
Figura 13 - Peça Web de “renderização” de <i>User Controls</i>	28
Figura 14 - Exemplo de interface de <i>External Data Exchange</i>	30
Figura 15 - Propriedades de Actividade de <i>Handle External Event</i> obtidas da interface de <i>External Data Exchange</i>	31
Figura 16 - Estado <i>workflow</i> com actividade de <i>EventDriven</i>	32
Figura 17 - Processo de <i>Help-Desk</i>	33
Figura 18 - <i>Workflow Help-desk</i>	34
Figura 19 - UI de Marcação de Férias	35
Figura 20 - <i>Workflow</i> de Marcação de Férias	36
Figura 21 – Exemplo de XML de <i>feature</i> de <i>SharePoint</i>	39
Figura 22 – Exemplo de XML de colunas	39
Figura 23 - Exemplo de XML de <i>business data catalog</i>	41
Figura 24 - Propriedades dos <i>templates</i>	42
Figura 25 - Manutenção de destinatários de ofício	44
Figura 26 - Notificação de novas alertas	45
Figura 27 - Listagem de alertas pessoais	45

1 Introdução

Designa-se como projecto um escrito composto pela descrição de um conjunto de actividades, devidamente inter-relacionadas e coordenadas, delineadas dentro de objectivos precisos, limites de tempo e de orçamento, que constituem uma obra a realizar. [13]

Este projecto surge no âmbito do curso de Mestrado Integrado em Engenharia Informática e Computação da Faculdade de Engenharia da Universidade do Porto. Foi realizado na instituição Companhia Portuguesa de Computadores Informática e Sistemas, S.A. (CPCis), no período compreendido entre 5 de Novembro de 2007 e 24 de Março de 2008.

1.1 O Projecto de Implementação da Intranet do InCI

O Instituto da Construção e do Imobiliário (InCI) é actualmente composto por um conjunto diversificado de plataformas tecnológicas. Estas plataformas, por si, remetem um conjunto de funcionalidades de negócio próprias da organização.

No entanto, quase sempre as funcionalidades existentes no sistema actuam de uma forma isolada, funcionam com *standards* próprios e não possuem uma forma expedita de adaptação às novas regras de negócio do InCI. Estas requerem um suporte tecnológico modular e tão padronizado quanto possível.

Adicionalmente, sempre que é necessário fazer a interligação entre duas plataformas surge a necessidade de implementar novamente um conjunto de operações de controlo para assegurar a correcta comunicação entre ambas, como verificações de segurança, auditoria, estatística de acesso, registo de excepções, entre outras.

Tendo em consideração as debilidades actuais e tendo como objectivos principais a reutilização e flexibilização das diversas funcionalidades de negócio o InCI decide que é altura de criar uma nova plataforma, denominada Plataforma de Serviços do InCI.

A Plataforma de Serviços, sendo uma plataforma colaborativa e de gestão de conteúdos, é um novo paradigma que se estima ser capaz de transformar a forma de trabalhar e conduzir o negócio através:

- da agilização dos seus processos e transacções de negócio;
- do aumento da produtividade dos seus colaboradores;
- do fortalecimento das relações com os seus clientes e parceiros.

As plataformas tecnológicas que compõem esta nova solução envolvem para além da área de intranet, onde está inserido este projecto, uma área de internet, que complementa a solução.

O projecto reportado neste documento surge inserido neste contexto, com a elaboração de várias funcionalidades para esta nova plataforma. É utilizada uma metodologia de desenvolvimento de software baseada numa arquitectura orientada a serviços, sendo esta opção justificada pela possibilidade de dinamização dos processos a incluir no projecto. Os serviços desenvolvidos no projecto são exteriorizados através de integração com a internet. O projecto consiste no desenvolvimento de um conjunto de funcionalidades que enumeradas nos pontos seguintes:

- Implementação de uma prova de conceito sobre a plataforma *Microsoft Windows Workflow Foundation* (WFF), que posteriormente serviu de base ao desenvolvimento de vários processos do InCI.
- Implementação do processo de Marcação de Férias da Direcção Administrativa, Financeira e de Recursos Humanos do InCI
- Implementação de funcionalidades de verificação de requisitos mínimos que é transversal a vários processos, tais como a Concessão de Alvará e de Baixa de Alvará do Departamento de Qualificação do InCI.
- Desenvolvimentos sobre a plataforma *Microsoft Office Sharepoint Server* (MOSS), construindo a Gestão Documental, onde residem todos os documentos que são criados e mantidos pelo InCI.
- Implementação de funcionalidades para geração automática de documentos, assinados digitalmente e com base em *templates*, utilizando a nova versão de documentos do *Microsoft Office Word 2007* (de *OpenXML*).
- Implementação de um módulo de criação e manutenção de alertas.

1.2 Organização e Temas Abordados no Presente Relatório

O presente relatório está estruturado em 5 capítulos que descrevem o projecto desenvolvido, incluindo 2 anexos que complementam a informação.

O capítulo inicial inclui uma introdução ao projecto, o seu âmbito e principais objectivos.

No segundo capítulo é apresentada uma descrição do problema, o planeamento efectuado, algumas notas sobre a equipa de trabalho e metodologia de desenvolvimento adoptada.

No terceiro capítulo é efectuada a revisão tecnológica do projecto. São analisados os processos de desenvolvimento de software recorrendo a uma arquitectura orientada a serviços, quais os produtos e tecnologias da solução que compõem a plataforma tecnológica. É ainda apresentado um estudo sobre a plataforma usada no desenvolvimento deste trabalho: *Windows Workflow Foundation*.

O quarto capítulo apresenta em maior detalhe a solução desenvolvida. Descreve os principais pontos relevantes da implementação do projecto, iniciando com a visão geral sobre a arquitectura lógica do projecto, seguindo-se a apresentação da forma como foi conseguida a ligação entre o ASP.NET o *Microsoft Office Sharepoint Server 2007* (MOSS). Este capítulo apresenta, ainda alguns detalhes da prova de conceito realizada na plataforma de *Windows Workflow Foundation*, e descreve a implementação das funcionalidades desenvolvidas para a nova Plataforma de Serviços no âmbito deste projecto.

No quinto capítulo são apresentadas as ilações finais da forma como decorreu o projecto, com um breve resumo das principais conclusões a retirar do projecto, as dificuldades encontras ao longo do seu desenvolvimento e as perspectivas de evolução futuras.

O Anexo A mostra o detalhe do planeamento do projecto.

O Anexo B apresenta os atributos de documentos que fazem parte das bibliotecas da Gestão Documental.

2 Análise do problema

Em consequência quer da crescente competitividade do mercado, quer da cada vez maior exigência e diversidade de requisitos dos clientes, as empresas vêem-se obrigadas a alargar capacidades, reduzir custos, e baixar tempos de resposta, disponibilizando aos seus clientes, parceiros, funcionários, e fornecedores, fácil acesso aos serviços que asseguram. Tipicamente, as aplicações que disponibilizam estes serviços têm de combinar sistemas de informação empresariais com novas funcionalidades de negócio. Para assegurar uma presença competitiva no mercado, tais serviços devem apresentar as propriedades seguintes [15]:

- *Alta disponibilidade*, para ir de encontro as necessidades do ambiente empresarial global de hoje em dia;
- *Segurança*, de forma a proteger a privacidade dos utilizadores e a integridade dos dados empresariais
- *Fiabilidade e escalabilidade*, de modo a garantir que as transacções acontecem com rapidez e precisão;

Este capítulo efectua uma descrição do projecto a desenvolver, enumerando as funcionalidades desenvolvidas pelo aluno e apresentando o processo de desenvolvimento adoptado.

2.1 Projecto InCI

2.1.1 Instituto da Construção e do Imobiliário

O Instituto da Construção e do Imobiliário, I.P. (InCI, I.P.) é a entidade reguladora do sector da construção e do imobiliário.

A esta entidade compete-lhe atribuir os títulos para o exercício das actividades reguladas, nomeadamente, Alvará de Construção, Título de Registo, Licença de Mediação Imobiliária e Inscrição de Angariador Imobiliário.

A sua actuação visa potenciar um mercado de construção e do imobiliário moderno e competitivo através de uma efectiva acção inspectiva e fiscalizadora.

2.1.2 Projecto a Desenvolver

O projecto de implementação da intranet do InCI envolve a construção de processos informatizados, levando à desmaterialização desses processos.

Por processo entende-se de um ciclo completo de acções que são executadas seguindo um fluxo, tipicamente formando um *workflow*, acções estas que culminam num resultado final (saída do processo).

O número elevado de processos a incluir no sistema implica que a equipa responsável pelo desenvolvimento do projecto seja numerosa, obrigando a uma gestão eficiente. Os processos que o InCI executa são críticos na sua forma de actuar, i.e. ser capaz de eficazmente gerar e manter a informação inerente a todo o processo, pois se essa informação fosse perdida ou alterada o sistema tornar-se-ia inconsistente. Perante estes requisitos, os processos estão desenvolvidos de forma a assegurar essa consistência, produzindo resultados fidedignos.

O projecto a que o aluno se propôs visava a construção de várias funcionalidades utilizando uma abordagem que assentava numa arquitectura orientada a serviços.

Dado que o sistema serve todas as áreas que o instituto dispõe, importa perceber onde é que o trabalho do aluno se enquadra. Os parágrafos seguintes descrevem as funcionalidades implementadas no âmbito do presente projecto.

Windows Workflow Foundation

O projecto inicia-se com um estudo sobre a plataforma do *Microsoft Windows Workflow Foundation* (WWF), sendo construída uma prova de conceito na plataforma. Os trabalhos envolvem análise e implementação de uma prova de conceito que serve de base à utilização do WWF no projecto, nomeadamente serviços transversais, a serem utilizados por toda a equipa, e implementação de um processo de *workflow* completo.

Processo na Direcção Administrativa, Financeira e de Recursos Humanos

Depois de terem sido obtidas as funcionalidades indispensáveis ao funcionamento do WWF, os trabalhos avançaram com a realização do processo de Marcação de Férias que se insere na Direcção Administrativa, Financeira e de Recursos Humanos (DAFRH).

Pretende-se que os colaboradores do InCI possam marcar as suas férias. O utilizador tem disponibilizado um número de dias de férias por ano que deverá marcar sobre um calendário. Com as devidas verificações dos dias introduzidos, e quando o utilizador assim entende, obedecendo às datas estipuladas pelo InCI, o processo é encaminhado para o superior hierárquico. Este novo utilizador deverá validar as férias dos seus colaboradores (aprovar ou rejeitar). Quando todos os seus colaboradores estiverem com as suas férias aprovadas, este utilizador poderá pedir a sua aprovação ao respectivo superior hierárquico. Esta ideologia funciona até as aprovações chegarem ao nível mais superior, através da reutilização do processo.

A elaboração desta funcionalidade inclui as tarefas de análise de requisitos, construção da interface gráfica, lógica de negócio, camada de acesso a dados e camada de dados. De notar que é elaborado um processo completo, de modo a despistar eventuais falhas na arquitectura, pois, posteriormente à conclusão desta funcionalidade o modo de funcionamento da equipa seria diferente, passando cada elemento a operar sobre camadas específicas (conforme é descrito em 2.2.2.)

Processo no Departamento de Qualificação

É também elaborado um conjunto de serviços, para a camada de lógica de negócio, que servem para processos do Departamento de Qualificação (DQ). As funcionalidades são de verificação de requisitos mínimos a determinados dados que são inseridos.

Estes serviços estão construídos de forma a poderem ser utilizados por vários processos do DQ, pois existe efectivamente a necessidade de serem reutilizados (e.g. processo de concessão de Alvará ou de Baixa de Alvará). A sua implementação inicia-se com uma análise de requisitos, sendo posteriormente realizada a sua implementação.

Gestão Documental

A gestão documental envolve um conjunto de bibliotecas de documentos que são criadas na instalação de uma *feature* de *Sharepoint*, construída para o efeito. Essas bibliotecas armazenam a documentação do InCI e diversos *templates*. Existem vários tipos de conteúdo com a associação de propriedades necessárias a cada caso. Esta funcionalidade é transversal a toda a solução, podendo ser utilizada por qualquer departamento ou processo.

Geração automática de documentos com base em templates

Um dos objectivos do InCI com esta nova plataforma é a desmaterialização de muitos dos seus processos. A possibilidade de geração automática de documentos é uma funcionalidade importante no preenchimento deste objectivo.

Uma outra das funcionalidades desenvolvidas é a geração automática de documentos, assinados digitalmente, com base em *templates*. Assim, usando os dados associados aos processos que estão armazenados em base de dados, é possível com base em templates criar documentos, que são assinados digitalmente com um certificado, para efeitos diversos, como por exemplo envio de notificações aos clientes do InCI.

Módulo de Alertas

O módulo de alertas consiste em um conjunto de funcionalidades que foram desenvolvidas para criar alertas e fazer a manutenção dos alertas pessoais, que são armazenados em lista de *Sharepoint*. Cada utilizador do sistema pode visualizar na página inicial uma notificação com os alertas novos. Está incluída uma hiperligação para a página de listagem de alertas pessoais, onde se podem executar algumas acções, tais como ler ou eliminar os alertas.

2.2 Processo de Desenvolvimento de Software

De seguida é apresentado o processo de desenvolvimento de software adoptado no presente projecto.

2.2.1 Planeamento

O planeamento de todo o projecto foi criado no início do projecto, no entanto devido a imprevisibilidade de algumas das funcionalidades que teriam de ser desenvolvidas, foi necessário reestruturar este planeamento. A abordagem inicial comportava uma visão inadequada sobre o projecto, descurando a necessidade da construção da base do projecto. Esta base tinha por objectivo permitir à equipa de desenvolvimento, numa fase mais avançada, a construção de serviços e funcionalidades de forma bastante mais célere.

Os planeamentos seguintes foram constituídos com uma visão mais reduzida sobre a quantidade de trabalho prevista, permitindo uma maior adequação às necessidades de desenvolvimento de cada funcionalidade. Esta vertente permitiu maior acerto e também a cada momento ser possível ter acesso a informação sobre o estado geral do projecto, podendo descer ao detalhe de cada funcionalidade.

A figura a seguir apresentada (Figura 1) espelha o planeamento que foi atribuído ao aluno. Verificando que as capacidades deste serviriam melhor em áreas de negócio transversais ao

projecto, abandonou a área de gestão administrativa e financeira do InCI, em que esteve inicialmente envolvido, pois, esta área foi remetida para segundo plano.

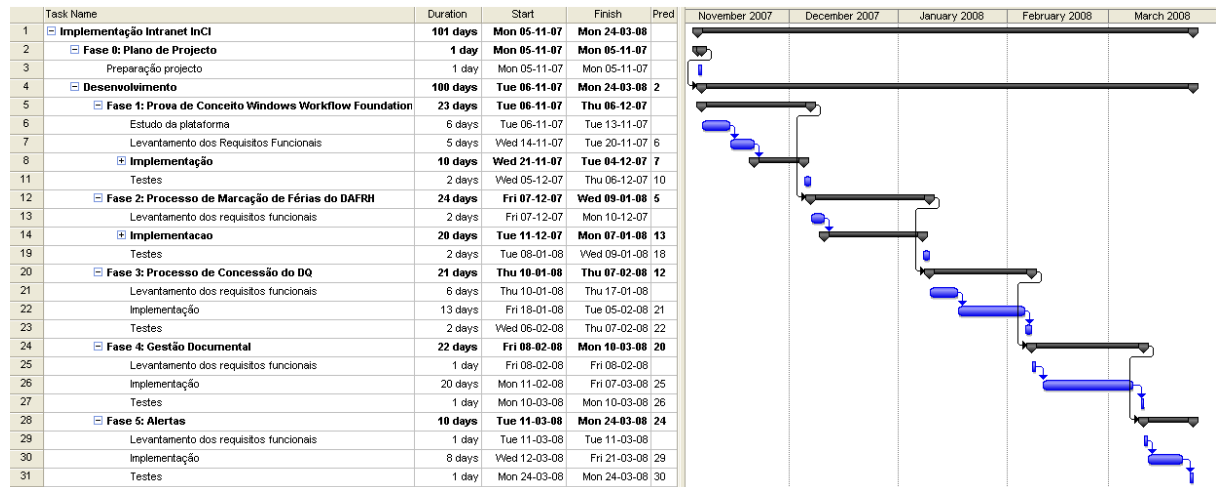


Figura 1 - Planeamento das actividades do projecto

2.2.2 Equipa de desenvolvimento

A equipa de desenvolvimento que constituiu este projecto é composta por 10 pessoas.

A equipa tem um gestor de projecto, e vários grupos divididos pelas áreas necessárias a desenvolver no projecto. Esses grupos são de duas ou três pessoas, e geralmente procuravam trabalhar em prol da elaboração da mesma funcionalidade ou grupo de funcionalidades. A esses elementos são atribuídas micro-tarefas a elaborar. Tipicamente estas são divididas pelos elementos de acordo com as camadas (ver 4.1) para as quais estão destacados para desenvolver.

O aluno desempenhou maioritariamente as funções de programador, com especial incidência nos serviços, e também de gestor de base de dados.

2.2.3 Metodologia

O processo de desenvolvimento de software utilizado neste projecto foi sempre sendo alvo de avaliações internas, de modo a perceber se estaria a ser o mais adequado, resultando por isso em algumas mudanças ao longo do projecto que são brevemente descritas de seguida.

Após a definição, pelo cliente, de que a CPCis seria a empresa a desenvolver o projecto, foi constituída uma equipa para a elaboração da análise da solução. Foi efectuada um levantamento inicial, servindo de base à análise detalhada obtida no momento de desenvolvimento das funcionalidades, realizando uma nova intervenção de análise, mais profunda, de forma obter toda a informação indispensável ao desenvolvimento. Esta metodologia levou a que o cliente, por vezes, efectuasse algumas alterações em relação ao que fora inicialmente previsto. Após a determinação de todos os requisitos necessários, por funcionalidade, passa para a equipa de programação a função de dar continuidade ao processo, com a construção da funcionalidade.

Quando esta nova fase termina, a funcionalidade é instalada na máquina de qualidade/testes. Esta máquina consiste em um servidor Windows Server 2003, com SQL Server 2005 e SharePoint Server 2007. Estes elementos constituem os principais produtos que são

necessários no projecto (ver detalhe em 3.3.1). Posteriormente à elaboração de testes sobre a funcionalidade, esta é colocada na máquina de pré-produção do cliente, permitindo-lhe a execução dos testes finais, de modo a ser dado por concluído o desenvolvimento de mais uma funcionalidade, com aceitação pelo cliente, ou no caso de haver um retorno negativo ser feita uma nova iteração sobre esta funcionalidade.

Embora esta metodologia não siga rigidamente nenhum modelo pré-concebido, tem algumas semelhanças com alguns dos modelos bem conhecidos, podendo mesmo ser feito a analogia com o modelo de *Staged Delivery*. A Figura 2 apresenta uma visão geral sobre este modelo de desenvolvimento.

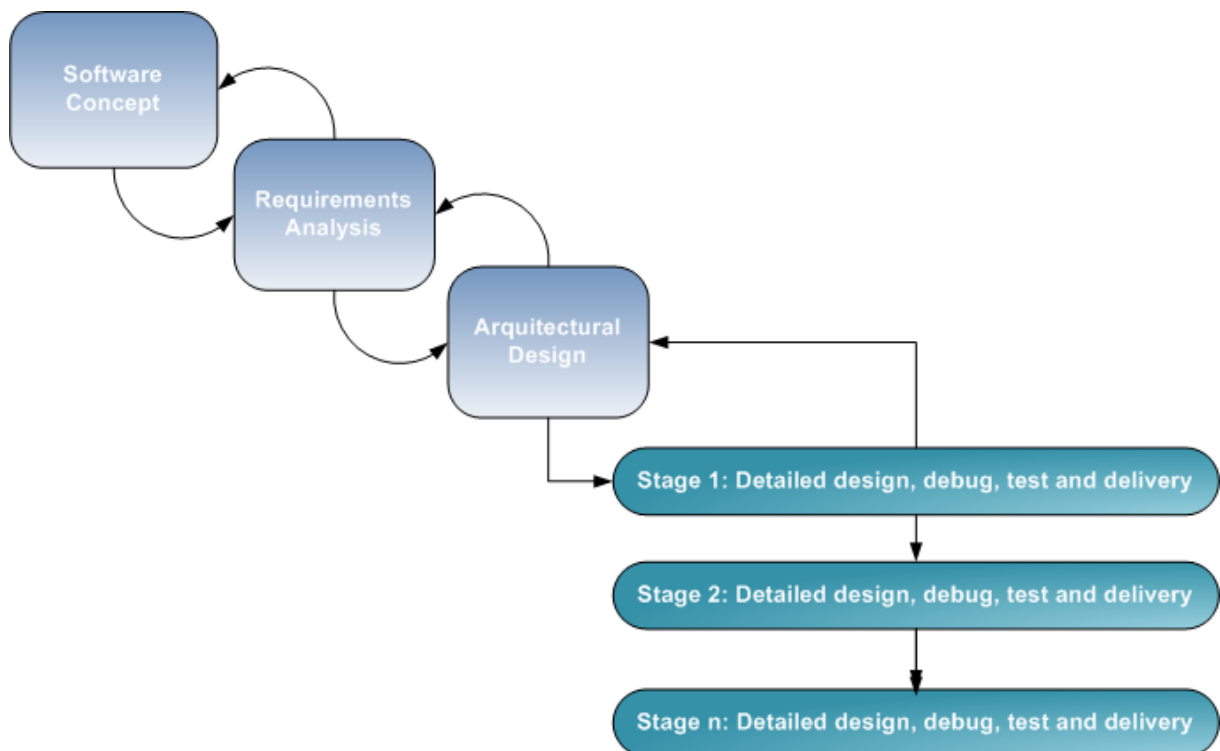


Figura 2 – Modelo de desenvolvimento de *Staged Delivery*

O *Staged Delivery* é um modelo que fornece o software em entregas faseadas. Este modelo tem como característica o desenvolvimento das principais funcionalidades da solução numa fase inicial do projecto, apresentando progressos incrementais em cada entrega ao cliente. Isto permite que o cliente dê um maior *feedback* à funcionalidade construída. [10]

O modelo *Staged Delivery* baseia-se no modelo de Cascata, seguindo-o na parte inicial onde é realizada a concepção, obtidos os requisitos e a arquitectura, após essas fases, o modelo prossegue para as fases que são compostas por uma análise detalhada, implementação e *debug*, testes e entrega ao cliente da funcionalidade completa.

3 Revisão Tecnológica

Neste capítulo é apresentado o modelo da Arquitectura Orientada a Serviços (SOA), pois esta é a abordagem adoptada para o desenvolvimento de software no actual trabalho. É também discutido o enquadramento com a plataforma tecnológica utilizada no projecto, incluindo uma análise ao *Windows Workflow Foundation*.

3.1 Arquitectura Orientada a Serviços

3.1.1 Introdução

A Arquitectura Orientada a Serviços (que em inglês é Service Oriented Architecture - SOA) [1] é um modelo de desenvolvimento de aplicações distribuídas. Os serviços são componentes distribuídos que fornecem interfaces para processar e entregar mensagens XML (eXtensible Markup Language). Um desenvolvimento baseado em serviços faz sentido para soluções transversais à organização, departamentos, e fronteiras entre as várias áreas da empresa. Um negócio com múltiplos sistemas e aplicações pode utilizar uma SOA para construir uma solução integrada, com funcionamento independente (*loosely coupled*), e que implementa fluxos de trabalho (*workflows*) unificados.

A Figura 3 apresenta o modelo de uma SOA.

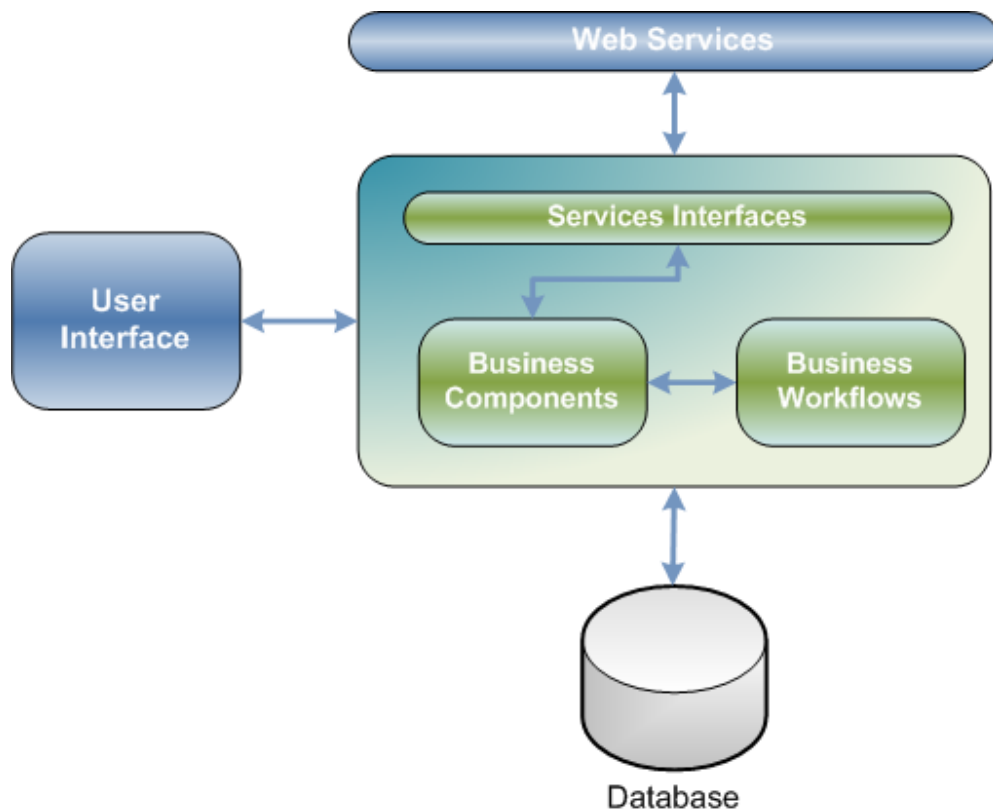


Figura 3 - Arquitectura Orientada a Serviços (SOA)

Uma SOA é composta por “serviços” que são unidades funcionais, representando as funcionalidades de negócio da organização, e que confinam a lógica de negócio e dados e expõem uma interface (*contract*), através da qual as aplicações podem fazer pedidos e receber respostas.

Essas aplicações não têm acesso à lógica de negócio ou dados de negócio directamente, sendo este acesso efectuado através da interface (*contract*). Tal facto dá aos programadores de *software* a possibilidade de poderem posteriormente alterar a lógica de negócio ou a base de dados (podendo, por exemplo, mudá-la de servidor), sem se preocupar com o impacto, que possa advir dessa operação, noutras aplicações.

3.1.2 Benefícios essenciais da SOA

Os parágrafos seguintes descrevem de forma resumida os benefícios mais relevantes da utilização da SOA

Redução de custos e tempo de implementação

Uma SOA permite reduzir custos e tempo de implementação de outros serviços ou unidades funcionais de negócio através da reutilização de outras funcionalidades já implementadas. Além disso, é esperado que haja um aumento da produtividade das equipas de desenvolvimento através da focalização destas no desenvolvimento no seu negócio (as equipas somente necessitam de publicar os “contratos de negócio” que os seus serviços de negócio disponibilizam).

Reutilização

A reutilização pode-se revelar vantajosa para a produtividade de uma organização produtora de software.

A maior parte das aproximações propostas até hoje, como *Component-Based Design*, falharam a promessa neste domínio. Sendo prematuro reclamar que uma SOA realiza essa promessa, estima-se que uma definição de serviços com granularidade menos fina fará com que a reutilização seja atingível a um nível mais satisfatório.

Agilidade

Agilidade representa a capacidade de resposta às mudanças que ocorrem inevitavelmente quer no ambiente quer nos requisitos de negócio. A independência de serviços e a separação de interface da implementação fazem com que a SOA seja um facilitador das exigências de agilidade empresarial.

Numa implementação com SOA, será possível alterar regras de negócio, realizar grandes mudanças na implementação ou mesmo mover um serviço de uma plataforma para outra com alterações mínimas ou até nulas nos restantes sistemas.

Funcionamento em ambiente heterogéneo

Grandes organizações têm, quase não excepcionalmente, uma variedade de ambientes de trabalho incluindo sistemas operativos, sistemas de desenvolvimento, plataformas, ou servidores aplicativos.

Estas diferenças tornam-se irrelevantes numa SOA quando utilizada com protocolos de *web services* como *HTTP* (*Hypertext Transfer Protocol*), *SOAP* (*Simple Object Access Protocol*) e

WSDL (Web Service Definition Language). O resultado é que os serviços podem ser construídos utilizando a plataforma mais apropriada para cada caso em vez de ser forçada a utilização de determinada plataforma por motivos de interoperabilidade.

3.1.3 Propriedades de um Serviço

Os serviços presentes numa arquitectura orientada a serviços apresentam um conjunto de propriedades inerentes a este modelo. Estas são descritas nos parágrafos seguintes.

Interface

A interface de um serviço proporciona o único mecanismo através do qual outros serviços ou aplicações externas podem aceder às operações ou métodos que compõem o serviço.

As informações na interface especificam os valores de retorno e os parâmetros dos métodos disponibilizados.

Os serviços devem ser projectados de forma a efectuar a maior quantidade de trabalho possível, procurando reduzir o número de chamadas a outros serviços (ao contrário do que acontece em programas orientados a objectos, onde deverá existir um maior número de chamadas atómicas), porque chamadas entre serviços são mais consumidoras de recursos do que chamadas típicas a métodos ou funções dentro de uma aplicação.

Apesar de não ser possível, normalmente, expressar tal facto na interface de um serviço, cada operação dispõe também de pré-condições, pós-condições e excepções. Estes itens, quando em conjunto com outros serviços que devam ser chamados numa ordem determinada para produzir um resultado satisfatório, constituem o que se denomina um contrato.

Implementação em “caixa preta”

Os serviços têm como objecto garantir a execução de regras de negócio e a manutenção de dados, tipicamente armazenados numa base de dados.

Esta conjugação de código, dados, produtos e *middleware* proporciona a funcionalidade do serviço e é designada no seu todo como implementação.

O serviço encapsula tudo isto e esconde-o atrás da interface, que é a única peça de que os consumidores do serviço têm conhecimento, o que leva a que esta seja chamada uma “caixa preta”.

A SOA não implica quaisquer restrições ou direcções sobre como construir os serviços, o que se revela vantajoso, permitindo às organizações implementá-los recorrendo-se das tecnologias, recursos e conhecimento existentes nos seus recursos humanos, podendo ainda utilizar técnicas como programação orientada a objectos e/ou reutilização de componentes.

Loose Coupling

Um dos principais objectivos da SOA é a independência na execução de serviços, isto é, um par de serviços deve ser capaz de comunicar com conhecimento mínimo acerca um do outro.

Para se invocar um serviço é estritamente necessário conhecer a sua interface e o contrato, mas inserido neste ambiente (*loosely-coupled*), o consumidor apenas se restringe a isso, não

conhecendo a sua localização, as regras de negócio que implementa, a linguagem em que foi programado, base de dados que utiliza ou sistema operativo onde assenta. Isto permite que possam ser efectuadas mudanças a qualquer um destes aspectos sem resultar num impacto para o cliente do serviço. Esta característica revela-se vantajosa em termos de manutenção, proporcionando uma flexibilidade acrescida.

A separação da interface de um serviço e da sua implementação proporcionam a base para a independência entre serviços.

Seguem-se mais algumas considerações sobre independência:

- Um serviço deverá chamar outro através da sua interface. Especificamente, um serviço não poderá aceder ao estado ou lógica de outro directamente (e.g. através de base de dados ou variáveis globais).
- Um serviço não deverá depender da existência de uma terceira parte, como um coordenador transaccional ou serviço de autenticação para que uma chamada tenha sucesso.
- Um serviço não deve exigir que os seus consumidores tenham invocado ou devam invocar seguidamente outros serviços para que haja sucesso.
- Um serviço não deve saber ou preocupar-se com quem o consome.

Modelo de programação sem estado (stateless)

Num modelo de programação orientado a objectos, um programa cria um objecto, após o que dispõe dessa instância até entregá-la a outra entidade ou até que a destrua.

A SOA pressupõe no seu ambiente que esta relação entre serviços seja realizada de forma diferente. Não existe o conceito de posse ou de instância. O que existe é uma única instância do serviço com capacidade de responder a múltiplos pedidos de consumidores.

Como resultado disto, um serviço não pode manter memória das chamadas anteriores de determinado consumidor, o que significa que um consumidor deve passar, com o seu pedido, informação sobre o contexto suficiente para ser identificável ou verificável.

Granularidade

Para que a SOA seja eficiente, os serviços devem ser definidos a um nível de granularidade não demasiado fino, caso contrário (com serviços que executem actividades muito atómicas), pode ser impossível garantir a independência desejada neste tipo de arquitectura e culminar mesmo em baixos níveis de desempenho. Se por outro lado, se analisar o extremo oposto (existência de serviços que executem actividades demasiado genéricas), verifica-se que os serviços disponibilizados servem para o contexto para o qual foram desenhados, impedindo a sua reutilização.

Esta análise permite concluir que algures no meio destes dois extremos se deverá encontrar a melhor opção, permitindo que um sistema complexo seja composto por vários serviços reutilizáveis e independentes.

Naturalmente, trata-se de uma avaliação subjectiva, sendo necessária experimentação e análise para “acertar” no equilíbrio adequado.

Desconfiança Saudável

Os serviços devem desconfiar de forma saudável do mundo exterior, ou seja, tudo o que não é a sua própria implementação. Este facto é essencial para atingir os objectivos de independência e reutilização.

Um serviço deve assumir que quaisquer dados recebidos podem ser maliciosos ou incorrectos. Consequentemente, todos os serviços devem incluir mecanismos de validação e autorização para garantir consistência de dados.

Um serviço não deve ser limitado a um grupo de consumidores nos quais “confia”, uma vez que ele não poderá ter a noção completa de quem o consumirá. Por outro lado, limitar a utilização de um serviço impossibilitará cenários de reutilização como, por exemplo, expô-lo na Internet a outra organização.

3.1.4 Princípios dos Serviços

No desenvolvimento de uma SOA, consideram-se quatro princípios essenciais: a explicitação das fronteiras, a autonomia dos serviços, a partilha de informação entre serviços e a compatibilidade de serviços. Estes princípios são, descritos sucintamente nos parágrafos seguintes

Princípio 1: As fronteiras são explícitas

As fronteiras, que são essencialmente as redes físicas, com todos os “pseudo-problemas” que daí possam advir, tais como a imposição de latências ou mesmo insegurança, devem ser consideradas explicitamente, ao invés do que anteriormente acontecia noutros tipos de abordagem, em que se procurava que a invocação de componentes fosse transparente à sua localização física. Por outro lado, as fronteiras procuram estar perfeitamente definidas formalmente e esconder as abstracções e conceitos utilizados internamente nos serviços.

O foco está na definição da fronteira e nas condições para a sua invocação e não na arbitrariedade da sua implementação interna, que se toma por certa.

Um exemplo da violação deste princípio é ter listas de parâmetros que recebem ou devolvem *DataSets* ADO.Net e uma forma de verificar se o mesmo é preservado, é verificar se a implementação de um serviço pode ser mudada para uma plataforma tecnológica, linguagem de programação e *middleware* completamente distintos, mantendo-se o *schema* da interface do serviço.

Princípio 2: Os serviços são autónomos

Os serviços interagem entre si utilizando mensagens (pedido-resposta), sem dependências implícitas. Como exemplo, a capacidade de um serviço que depende de outros, deve funcionar mesmo na situação em que esses outros serviços não estão disponíveis.

Princípio 3: Os Serviços partilham Esquemas e Contratos, não classes

A invocação de um serviço deve apenas necessitar de conhecer qual o seu *Schema* (Esquema) e o contrato. Estes definem respectivamente, a estrutura de invocação de um serviço, e qual o seu comportamento. Não é necessário para o cliente ter qualquer outro conhecimento relativamente ao serviço.

Note-se que estes dois requisitos podem ser expressos de forma independente da tecnologia usada na implementação (como *Schemas XSD*). A utilização de informação “fora-de-banda” (como cabeçalhos SOAP) para passar contextos de transacções através de fronteiras de serviços, é uma violação deste princípio.

Princípio 4: A compatibilidade entre serviços é baseada em Políticas

Uma *policy* (política) é constituída por uma descrição das capacidades oferecidas por um serviço e pelos requisitos necessários para a sua invocação.

Um exemplo de *policy* é a exigência de que na invocação de um serviço sejam passados *tokens* seguros para autenticação.

A área de *policies* está ainda embrionária, tanto em termos de definição como de existência de mecanismos concretos para a definição de *policies*, prevendo-se para já a utilização do *Ws-Policy*, padronizado pelo Web Services Interoperability Organization (WS-I).

3.1.5 Tipificação de Serviços

Os serviços são tipicamente categorizáveis em 4 tipos, o que facilita a sua estruturação em termos tecnológicos e possíveis formas de composição.

- **Serviços de Processo:** representam processos de negócio, normalmente de longa duração, que podem envolver *workflows* complexos e interacção humana;
- **Serviços de Actividade:** coordenam operações de diversos Serviços-Entidade numa operação atómica.
- **Serviços de Entidade:** representam operações atómicas simples sobre entidades persistentes (e.g. Clientes). Podem escrever directamente em repositórios (e.g. bases de dados), invocar componentes, aceder a sistemas legados, ou até invocar serviços em parceiros externos.
- **Serviços de Infra-estrutura:** Implementam processos não funcionais e transversais aos restantes serviços. (e.g. autenticação e segurança, *logging*, gestão e monitoria).

3.1.6 Peças constituintes da SOA

Numa arquitectura SOA, destacam-se três grandes blocos:

- *Service Consumers*;
- *Service Bus*;
- *Service Providers*.

Service Consumer

O *Service Consumer* é um módulo que interliga e controla “serviços”, através de interacções com os utilizadores e processos de *workflow*.

O *Service Consumer* não se preocupa com a implementação da funcionalidade que pretende, simplesmente tem em consideração os contratos publicados pelos *Service Providers*, a nível de:

- Mensagem;
- QoS (*Quality of Service*);
- Capacidades requeridas.

Service Bus

O *Service Bus* é uma unidade conceptual que permite aceder à SOA. Pode apresentar-se numa forma centralizada, baseando-se num aglomerado de serviços, ou apresentar-se de forma distribuída ou descentralizada.

Tipicamente, é construído em cima de um modelo *pipeline* (Figura 4).

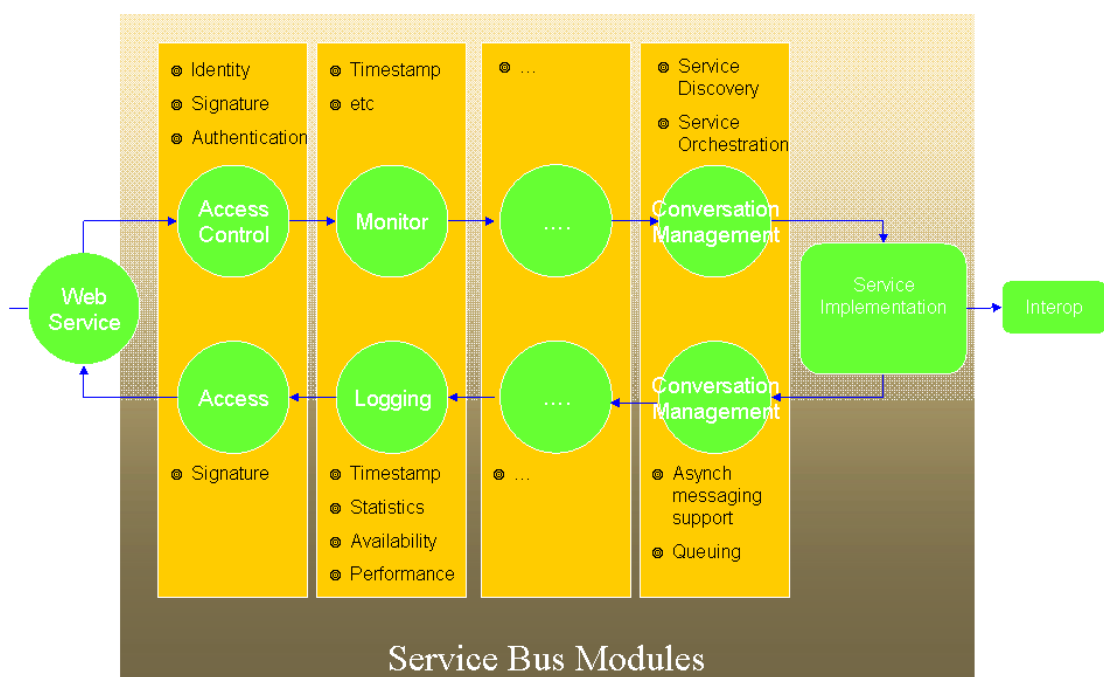


Figura 4 – Modelo de *Pipeline* do *Service Bus* [2]

O *Service Bus* mantém um directório de todos os “serviços” existentes na arquitectura, e é responsável por assegurar qualidade de serviço (QoS). Contém ainda funcionalidades para realizar o encaminhamento de mensagens, bem como assegurar a segurança e monitorização no envio de tais mensagens.

Service Provider

O *Service Provider* é uma funcionalidade de negócio ou de infra-estrutura. O contrato e mensagens de comunicação são bem conhecidos. Apresenta capacidades transaccionais, assegurando que várias operações de negócio funcionem como um todo, garantindo sucesso ou insucesso global, e portanto preservando a integridade do *workflow* total. Os pormenores de implementação são desconhecidos para o *Service Consumer*.

3.2 Funcionamento da Arquitectura Orientada a Serviços

No funcionamento de uma aplicação SOA torna-se importante perceber o processo de implementação de um serviço, e o modo como são realizadas a comunicação inter-serviços e a gestão de excepções.

3.2.1 Fases da Implementação de um Serviço

A utilização da rede de serviços da plataforma SOA pressupõe essencialmente quatro fases:

- Registo e Credenciação (*Service Provider* e *Service Consumer*);
- Desenvolvimento do *Service Provider*;
- Publicação do “contrato” do *Service Provider* no directório de serviços;
- Utilização do *Service Provider* pelo *Service Consumer*.

Registo e Credenciação

Esta fase é caracterizada pela obtenção das credenciais de autenticação de acesso à rede de serviços (Figura 5).



Figura 5 - Registo e credenciação de um serviço

O *Service Consumer* ou *Provider* contacta a “Entidade Responsável da rede de serviços” para realizar o registo, obtendo como resposta as credencias. As credenciais poderão ser do tipo:

- Par nome de utilizador/palavra-chave;
- Certificado Digital X.509v3;
- Identidade Windows.

Desenvolvimento do Service Provider

Nesta fase desenvolvem-se as funcionalidades que se pretendem incluir no *Service Provider*, que deverão obedecer aos princípios básicos já apresentados nas secções anteriores e serem expostas segundo um “contrato/interface”.

Publicação do “contrato” do Service Provider no directório de serviços

Após o término da fase anterior, com subsequente estabilização das funcionalidades, deverão estas passar a constar do directório de serviços. O *Service Provider* publica o serviço através de uma interface Web no directório de serviços.

Utilização do Service Provider pelo Service Consumer

Após os serviços serem publicados, podem estes ser usados pelo *Service Consumer*. Este processo é apresentado na Figura 6.

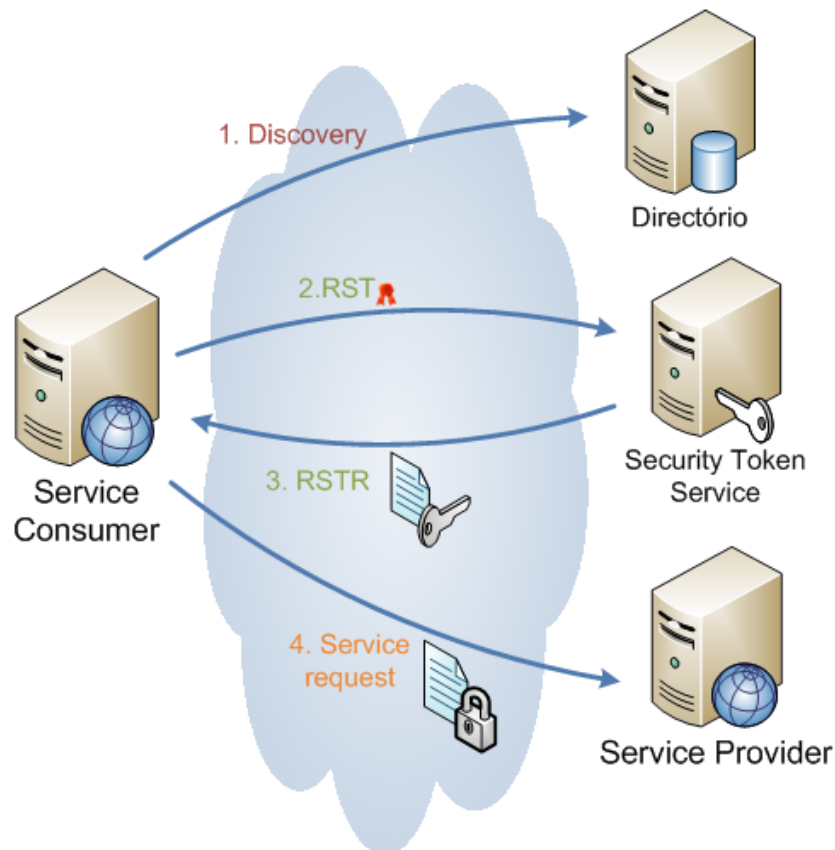


Figura 6 - Consumo de um *service provider*

A utilização do serviço desenvolvido é um procedimento que inclui os seguintes passos:

1. Consulta do directório de serviços para obtenção do contrato e localização do serviço a consumir (*Service Discovery*);
2. Pedido de um *token* (RST) válido na rede de serviços, emitido pelo *Security Token Service* (STS). O pedido do *token* deverá ser acompanhado pelas credenciais obtidas no processo de Registo. As credenciais serão validadas pelo STS;
3. Obtenção do *token* (RSTR) válido na rede de serviços;
4. Finalmente, é efectuado um pedido (mensagem) de consumo de serviço ao *Service Provider*, mensagem esta que inclui também o *token* obtido.

3.2.2 Comunicação Inter serviços

A comunicação entre os serviços, que constituem a plataforma, é baseada em mensagens XML: SOAP (*Simple Object Access Protocol*) Envelopes. As mensagens SOAP Envelopes são constituídas por dois campos: *Header* e *Body*, que significam, respectivamente, cabeçalho da mensagem e corpo da mensagem.

SOAP Envelopes são documentos XML, o que possibilita que a utilização dos protocolos de interoperabilidade entre os *web services* (WS-I) seja viável. Alguns exemplos de protocolos comportados são:

- *WS-Security* – Integra um conjunto de tecnologias de segurança, incluindo assinaturas digitais e encriptação baseada em *tokens* seguros;
- *WS-Trust* & *WS-SecureConversation* – Estabelece uma comunicação segura orientada a sessões, utilizando *tokens* seguros;
- *WS-Addressing* – Identifica pontos de destino do serviço nas mensagens e permite que esses pontos se mantenham actualizados à medida que uma mensagem é passada por dois ou mais serviços.

3.2.3 Gestão de Excepções

A gestão de excepções dos serviços é concretizada através da utilização dos serviços transversais da plataforma de *exception management*.

A interoperabilidade com os diversos *Service Providers* da plataforma poderá ser efectuada através de duas aproximações:

- a) Utilização directa¹ das classes .Net que implementam as funcionalidades pretendidas;
- b) Troca de mensagens contrato (no caso de *web services*).

A Figura 7 apresenta um modelo de consumo de serviço de negócio.

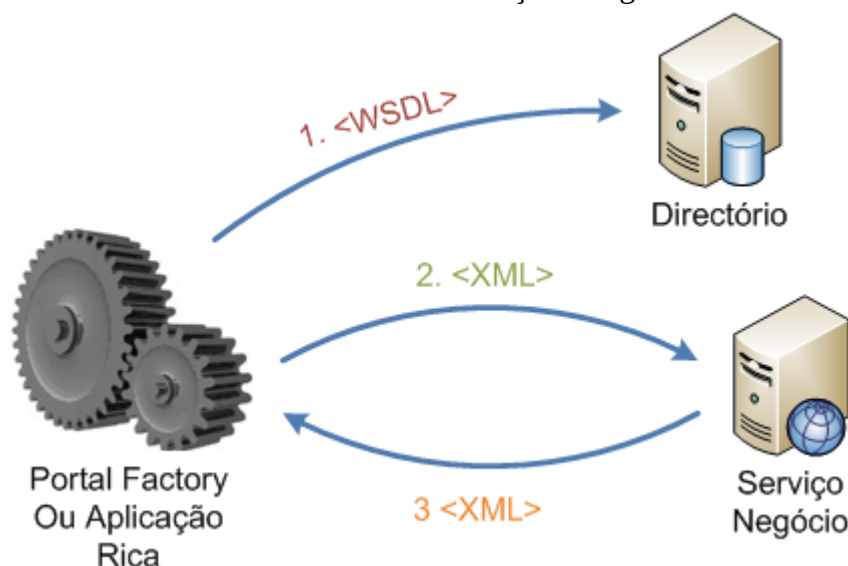


Figura 7 - Modelo de consumo de serviço de negócio

De uma forma dinâmica, os *Service Consumers* conseguem obter os contratos ou interfaces

¹ Residentes na mesma infra-estrutura do serviço.

O serviço de negócio é invocado tendo em consideração as suas *policies*² e o seu interface.

A resposta é apresentada ao utilizador final tendo em consideração o seu contexto (“*Portal factory*” ou Aplicação Rica).

3.3 Plataforma tecnológica

Esta secção apresenta a arquitectura tecnológica utilizada no desenvolvimento do portal da intranet do InCI.

Convém aqui referir que a opção de escolha de toda a plataforma tecnológica, incluindo o WWF (que será apresentado em 3.5), parte de um cadernos de encargos que a Microsoft Portugal, que é a empresa consultora do projecto, forneceu ao InCI. Assim, as escolhas tecnológicas do projecto, e consequentemente do aluno, estavam previamente definidas.

3.3.1 Produtos e Tecnologias da Solução

A seguinte figura (Figura 8) ilustra quais os produtos e tecnologias da solução que compõem a plataforma tecnológica:

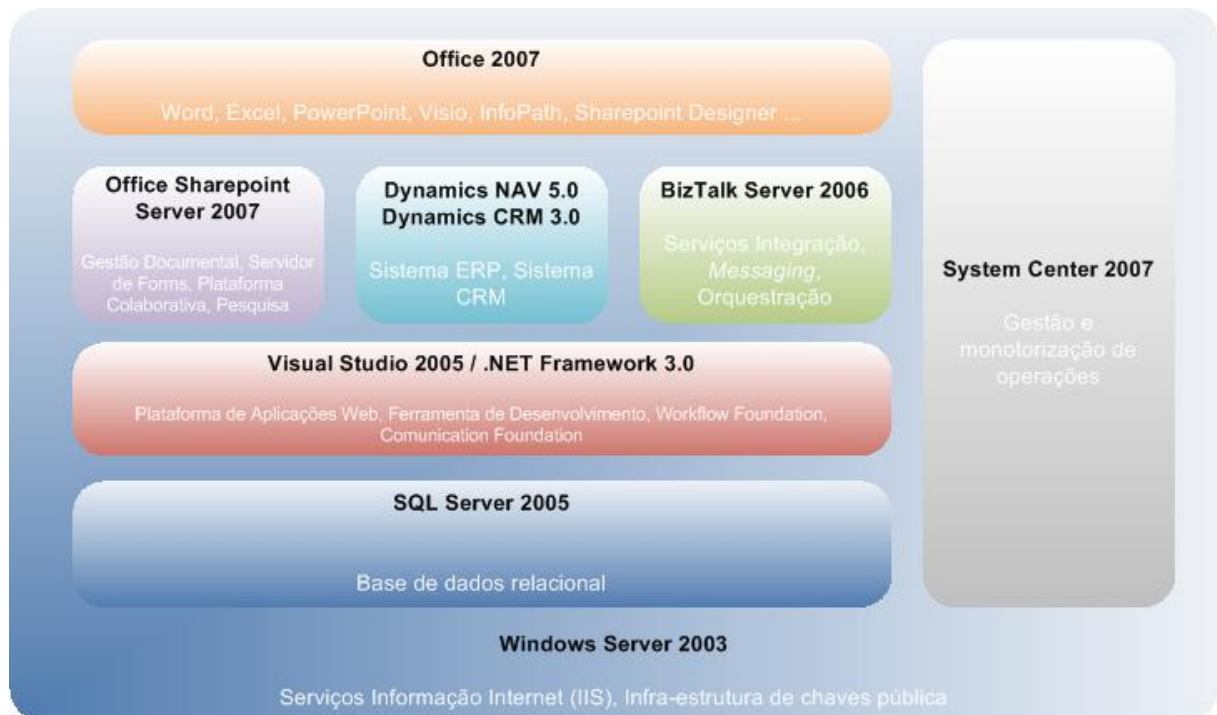


Figura 8 - Produtos e Tecnologias da Solução

Os seguintes parágrafos explicam qual o papel que os principais produtos ocupam no seio do desenvolvimento do projecto.

² Políticas de invocação do serviço de negócio: tipo de autenticação, cifrar da mensagem ou de partes da mensagem, etc.

Windows Server 2003

O Windows Server 2003 é o sistema operativo utilizado para suportar todos os restantes produtos que integram a solução, fornecendo garantias de fiabilidade, disponibilidade, segurança e escalabilidade.

O Windows Server 2003 é ainda responsável pelos seguintes serviços (estando estes distribuídos por vários servidores):

- *Web Server e Web Application Services.*
- *Directory services*
- *Domain Name System (DNS)*
- *Dynamic Host Configuration Protocol (DHCP) server*
- *Windows Internet Naming Service (WINS)*
- *Public Key Infrastructure*
- *Windows Software Update Services (WSUS)*

.NET Framework 3.0 SP1

A *Framework 3.0* com *Service Pack 1* do .Net é a plataforma tecnológica utilizada para desenvolver todo o núcleo de funcionalidades.

A necessidade desta versão deve-se à inclusão de subsistemas como o *Workflow Foundation* e funcionalidades de geração automática de documentos com assinatura digital.

SQL Server 2005

O Sql Server 2005 é o repositório de dados a usar. Este é um sistema de gestão de base de dados (SGDB) que é capaz de funcionar em múltiplas situações, assegurando o suporte de informação de forma segura e fiável.

Office SharePoint Server 2007

O *Microsoft Office Sharepoint Server (MOSS)* 2007 tem um papel central nesta solução.

Trata-se de um novo produto com diversas capacidades projectadas para suportar uma vasta variedade de funcionalidades empresariais para gerir conteúdos, apresentar dados de negócio, automatizar processos de *workflow*, gerir registos, e construir Web sites públicos e privados.

A Figura 9 ilustra as seis áreas operacionais que compõem o MOSS, que são a seguir descritas de forma resumida.

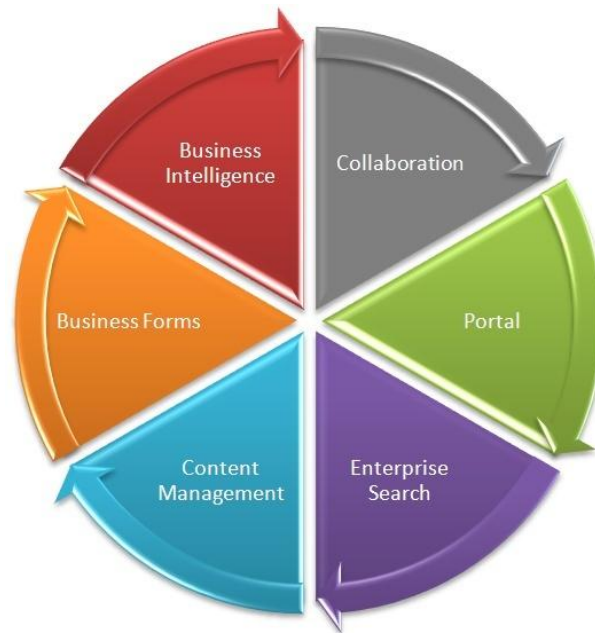


Figura 9 - Componentes do Microsoft Office Sharepoint 2007 [2]

Colaboração (Collaboration)

Os *Windows SharePoint Services* (WSS) desempenham um papel fundamental no MOSS, facilitando o acesso a contactos, documentos e conteúdos partilhados. O uso adequado destas funcionalidades permite aumentar a produtividade do trabalho em equipa e possibilita a tomada de decisões de forma bem informadas.

O WSS inclui suporte nativo para gestão de listas de contactos, eventos, tarefas, documentos, imagens ou listas *ad-hoc* definidas pelo gestor de conteúdos de cada portal.

No âmbito da colaboração, o MOSS inclui funcionalidades como gestão de versões de documentos, controlo de acesso para edição ou a integração com o Microsoft Office que permite, por exemplo, a um editor publicar um documento numa biblioteca em *SharePoint* directamente a partir da aplicação *Office*.

Portal

O MOSS inclui funcionalidades que simplificam a concepção e implementação de portais, com recurso a uma galeria de *templates* cobrindo diferentes cenários que vão desde os espaços colaborativos para pequenas equipas de trabalho até portais departamentais agregadores ou portais Internet. Existem ainda outras funcionalidades descritas a seguir:

- Audiências – possibilidade de definir destinos de conteúdos para utilizadores. No MOSS é possível definir audiências baseadas em utilizadores e grupos de *Active Directory* (AD). É possível também mostrar ou esconder conteúdos de páginas por audiência.
- Perfis (*Profile*) – os perfis de MOSS de utilizadores permitem definir, pesquisar, e apresentar metadados acerca de utilizadores do portal. Os perfis contêm tipicamente a informação base (e.g. Nome) da AD, sendo possível estender os “meta-dados” que o identificam com campos específicos, como por exemplo, línguas que o utilizador fala ou áreas de especialidade.

- “Meu Site” (My Site) - todos os utilizadores podem ter e gerir o seu próprio local (site) pessoal. Este local funciona como o ambiente de trabalho pessoal, permitindo organizar documentos e informações.
- *Single Sign-On* (SSO) – No MOSS o SSO facilita o acesso a várias outras funcionalidades, como por exemplo dados noutros sistemas de negócio. O SSO permite especificar um conjunto de credenciais que são utilizadas para aceder a sistemas secundários, fornecendo a essas credenciais um nome aplicacional que poderá ser invocado quer por funcionalidades do MOSS quer por outro tipo de código.

Pesquisa (Enterprise Search)

O componente de indexação e pesquisa do MOSS proporciona uma experiência de utilização consistente e familiar, fornecendo resultados precisos e ordenados de acordo com a sua relevância. A pesquisa pode incidir não apenas sobre documentos e conteúdos Web, mas também contactos e dados mantidos em aplicações externas, facilitando o acesso transversal a toda a informação. O serviço é flexível para permitir a indexação e pesquisa por “meta-dados” específicos expostos pelas páginas Web ou documentos Office, como por exemplo o autor ou tema atribuído a um conteúdo.

Gestão de Conteúdos (Content Management)

As funcionalidades de ECM (*Enterprise Content Management*) são um termo que se refere à gestão de todo o ciclo de vida de um documento, desde a sua criação até ao seu arquivamento. Isto inclui não só as capacidades de *workflow*, mas também a possibilidade de arquivar estes registos e aplicar políticas de retenção sobre eles.

O ECM inclui adicionalmente a gestão de conteúdos Web, *Web Content Management* (WCM), para a criação, aprovação e publicação de *sites* Web.

O MOSS suporta diversos *templates*, permitindo às organizações criarem *sites* públicos e privados. Um exemplo desses *templates* permite criar um repositório de registos que pode ser utilizado para arquivar os documentos de acordo com uma política de retenção de documentos.

Formulários (Business Forms)

O serviço *Forms Server* permite a publicação em páginas Web de formulários desenhados recorrendo ao *Microsoft Office InfoPath 2007*. Utilizando estes formulários a informação pode ser submetida e armazenada de forma estruturada em documentos XML. O MOSS permite associar *workflows* à submissão dos dados do formulário, constituindo processos de negócio completos. A lógica associada aos *workflows* pode ser mais simples ou mais complexa, como por exemplo, e respectivamente, validação automática ou manual dos dados, ou interacção com repositórios de dados específicos ou outros serviços externos.

Business Intelligence

O MOSS oferece capacidades de *Business Intelligence* (BI) capazes de suportar decisões de negócio. As funcionalidades de BI do MOSS permitem desenvolvimento simplificado de

dashboards Web incorporando dados oriundos de fontes como *KPIs* (*Key Point Indicators*), Partes Web (*Web Parts*) ou folhas de cálculo Excel.

3.4 Níveis de Serviço da Solução

A solução desenvolvida deve ser modular e conter níveis de serviços adequados no que diz respeito a:

- Escalabilidade;
- Disponibilidade;
- Segurança.

Escalabilidade

A solução comporta a possibilidade de escalabilidade lógica e física. Nomeadamente, escalabilidade horizontal (*scale-out*), com a possibilidade de aumentar o número de servidores que são utilizados na camada de apresentação. E também em termos de escalabilidade vertical (*scale-up*), com por exemplo, o aumento da capacidade física dos servidores, acrescentando processadores, memória ou discos.

Disponibilidade

Uma solução como a que foi desenvolvida requer altos níveis de disponibilidade. A própria arquitectura física teria de ser dotada desses mesmos requisitos, através de uma *web farm* (com vários servidores, que servem de recurso mútuo no caso de algum deles falhar) e de um *cluster* de dois servidores de SQL Server.

Segurança

A solução está exposta a diversos canais, ambientes e utilizadores sendo indispensável que garanta níveis de segurança elevados. A segurança está assegurada pelo sistema operativo (Windows Server 2003) pela minimização dos serviços disponibilizados até ao indispensável, por uma maior protecção dos recursos e informação armazenados, fecho de protocolos, serviços e respectivos portos de acesso nas comunicações com a rede, e finalmente a activação de mecanismos de auditoria de todos os eventos e acções relevantes nos sistemas.

3.5 Windows Workflow Foundation

O *Windows Workflow Foundation* (WWF) é parte integrante da solução desenvolvida e surge como resposta a uma necessidade de controlo dos processos do InCI. Os seguintes parágrafos apresentam brevemente o conceito de *workflow* e a plataforma WWF.

Porquê Workflows?

Dependendo da natureza do negócio ou da complexidade do problema a resolver, os programadores têm de encontrar soluções para problemas de negócio reais. Independentemente do problema que surja, o ser humano tem a tendência de resolver os

problemas separando-o em pequenas partes. Essas partes são posteriormente divididas em tarefas mais pequenas, e assim sucessivamente.

Após a obtenção destas pequenas tarefas, estas são decompostas em passos necessários para realizar tais tarefas. Estes passos têm normalmente uma ordem associada, representando instruções individuais que só fazem sentido se forem executadas pela ordem correcta.

Um *Workflow* é simplesmente um conjunto de passos que são executados em uma determinada sequência para atingir um propósito definido de acordo com um conjunto de regras.[3]

Porquê Windows Workflow Foundation?

A Microsoft criou o *WWF* para simplificar o desenvolvimento de aplicações .NET que tenham *workflows* inseridos nos processos. O *WWF* possibilita a organização das regras de negócio. O *WWF* está dividido em dois tipos de *workflows*, os de estado e os de sequência. O primeiro é habitualmente mais adequado a *workflows* que tenham interacção humana. No caso de se pretender aplicações altamente configuráveis, poderá separar-se a lógica de negócio do fluxo de execução, levando a que se posteriormente for necessário efectuar alguma alteração ao fluxo, esta seja realizada sem alterações à lógica de negócio.

As principais vantagens da utilização do *WWF* podem ser enumeradas nos pontos seguintes [3]:

- Fornece uma plataforma flexível e robusta de desenvolvimento de *workflows*, poupando tempo no desenvolvimento “manual” de uma plataforma semelhante;
- Permite o desenvolvimento de aplicações de forma consistente. Um *workflow* assemelha-se muito com outro. Esta consistência no modelo de programação e ferramentas disponibilizadas melhoram a produtividade quando estão a ser desenvolvidas novas aplicações e na manutenção de aplicações existentes;
- Suporta *workflows* de sequência e *workflows* de máquinas de estado. Os *workflows* de sequência são tipicamente utilizados em interacções de sistemas. Os *workflows* de máquinas de estado são adequados na resolução de problemas de interacção humana;
- Suporta persistência de *workflows*. A possibilidade de guardar e depois obter o estado de um *workflow* em execução é especialmente importante quando se modelam interacções humanas, que podem levar a que um *workflow* seja executado durante várias horas ou dias.
- Fornece um ecossistema de *workflow* completo. Adicionalmente ao *workflow runtime*, a Microsoft fornece também um conjunto de actividades padrão, persistência de *workflows*, monitorização de *workflows* e *tracking*, e um construtor visual de *workflows* integrado no Visual Studio.
- É grátis e existe uma comunidade já numerosa de adeptos da tecnologia que partilham as suas ideias, os seus componentes com actividades, e outro código.

Tipos de Workflows

A seguir são apresentados em maior pormenor os dois tipos de *Workflow* já referidos: *Workflow* de sequência e *Workflow* de máquina de estado

Workflows de Sequência

O *workflow* de sequência é provavelmente o tipo de *workflow* mais comum. Este tipo de *workflows* descreve um processo que tem um ponto de início, realiza um conjunto de acções numa dada ordem, e depois culmina num estado final [14]. Um exemplo de um *workflow* de sequência é apresentado na Figura 10.

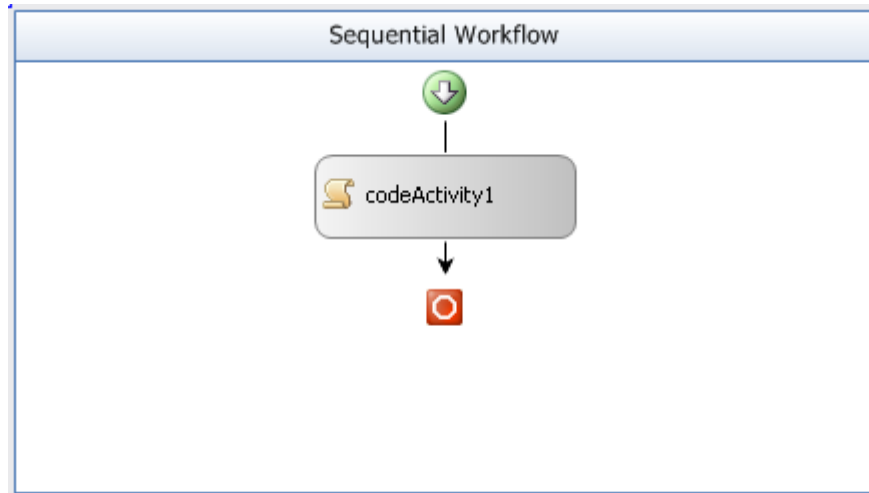


Figura 10 - Exemplo de *workflow* sequencial

No *workflow* de sequência, pode-se utilizar vários controlos de lógica que existem igualmente na programação tradicional, tais como *if-else* e ciclos *while*. A diferença entra as duas abordagens é que estes controlos são no *Workflow* definidos visualmente em vez de escritos numa linguagem de programação.

Workflow de máquina de estado

O *workflow* de máquina de estado não define uma sequência fixa de passos, mas em vez disso, define um conjunto de estados, com a indicação das possíveis transacções entre estados. Cada estado pode conter mais do que um passo que é executado durante a transacção do estado [3]. A Figura 11 ilustra um *workflow* do tipo máquina de estado:

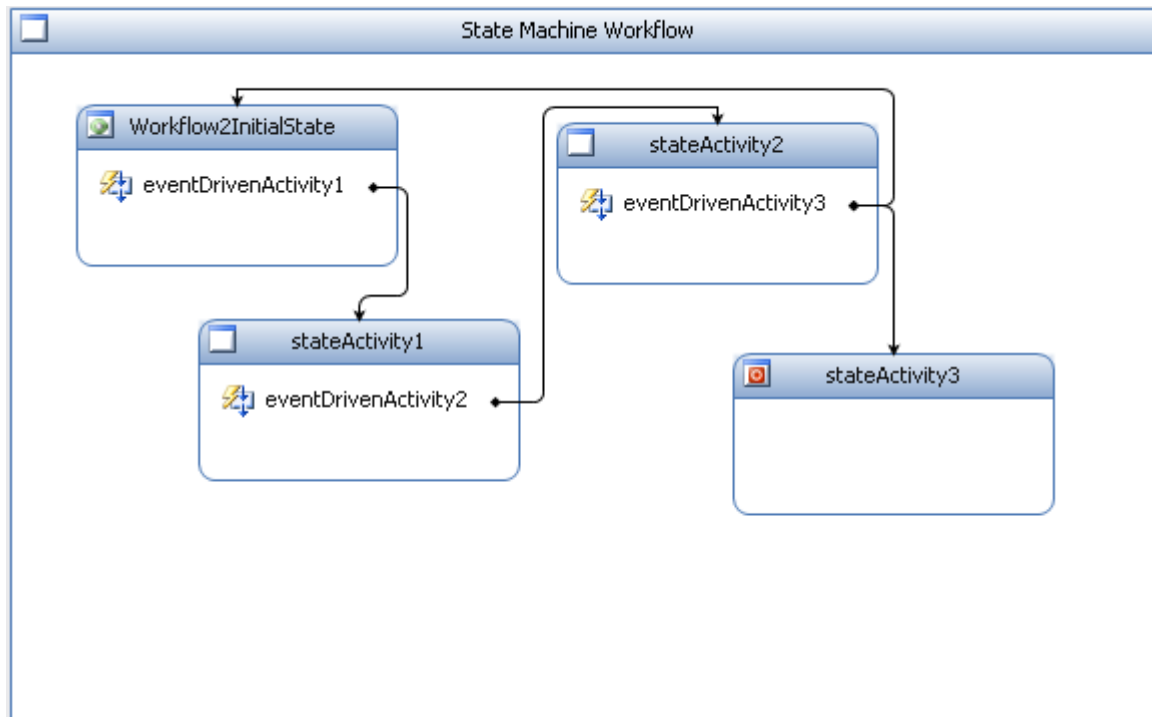


Figura 11 - Exemplo de *workflow* de máquina de estado

Um *workflow* de máquina de estado não está sujeito a uma sequência estática de passos. A execução não tem sempre de começar com tarefas no primeiro estado. Isto permite que o *workflow* possa ser interrompido e resumido se necessário. No *workflow*, as transacções entre estados são despoletados por eventos externos que são disparados pela aplicação. Isto significa que o controlo, em geral, é externo ao *workflow*.

Tipos de Serviços

A plataforma do WWF disponibiliza um conjunto de serviços *core*, que são brevemente apresentados[3]:

- *Scheduling* – Cria e gere os *threads* utilizados pelo motor de runtime que executa a instância de *workflow*;
- *Commit Work Batch* – Gere as transacções que são utilizadas pelo motor de runtime para manter a consistência entre o estado dos *workflows* internos e armazéns de dados externos;
- *Persistence* – Manuseia a persistência do *workflow*;
- *Tracking* – Fornece a capacidade de registar o eventos que são executados pelo *workflow*.

Os dois primeiros serviços são têm de se incluir sempre que se use WWF, enquanto que os restantes dois são opcionais.

4 Desenvolvimento do Projecto

Este capítulo explica como foi realizado o processo de desenvolvimento do projecto, apresentando a arquitectura lógica utilizada, a prova de conceito efectuada no *Windows Workflow Foundation* e as funcionalidades criadas, nomeadamente o Processo de Marcação de Férias da Direcção Administrativa, Financeira e de Recursos Humanos, o Processo de Concessão do Departamento de Qualificação, o módulo de Gestão Documental, o módulo de Geração Automática de Documentos e o módulo de Alertas.

4.1 Arquitectura lógica do projecto

A nova plataforma do InCI disponibiliza um conjunto de infra-estruturas tecnológicas e serviços aplicacionais, que respeitam o princípio associado ao desenvolvimento de software utilizando SOA. A Figura 12 demonstra de que formas estes conceitos estão agregados na arquitectura desenvolvida:



Figura 12 - Arquitectura lógica do projecto

Esta plataforma assenta num modelo distribuído de serviços que permite criar níveis de abstracção entre as diversas componentes aplicacionais, facilitando a implementação faseada da solução e a sua evolução futura. Os serviços presentes na plataforma podem ser agrupados em cinco áreas de acordo com as suas funcionalidades:

- **Serviços de Apresentação.** Portais ou aplicações ricas expostas nos diversos canais da plataforma que consomem os serviços de negócio (*service providers*) de acordo com as especificações publicadas ou por acesso directo a eles.
- **Serviços de Negócio.** Conjunto de serviços que serão consumidos pelos *service consumers* devidamente credenciados. Alguns exemplos destes serviços são o serviço de pesquisa, serviços de gestão documental ou serviços de negócio específicos.

- **Serviços Transversais de Suporte** (*base services*). Serviços de suporte utilizados por diversos outros serviços da plataforma (*service consumers* e *service providers*). Nesta categoria, incluem-se os serviços de segurança (autenticação, autorização, comunicação segura), directório de utilizadores e serviços, auditoria e *logging*, tratamento de excepções, gestão de configurações ou *workflow*.
- **Serviços de Dados**. Incluem um conjunto de funcionalidades para armazenar, recuperar, gerir e relacionar informação.
- **Serviços de Integração**. Serviços que efectuam a integração aplicacional com as entidades externas e aplicações internas. Adicionalmente, este módulo disponibiliza mecanismos de orquestração de alto nível de processos de negócio entre as diversas aplicações.

4.2 ASP.NET e Microsoft Office SharePoint Server

O projecto, na sua parte de núcleo de negócio, foi desenvolvido com base em formulários ASP.NET assentes em *Microsoft Office SharePoint Server* (MOSS).

Todos os formulários desenvolvidos no projecto, e que integram nos Serviços de Apresentação, são criados em *User Controls*, que são implementados da mesma forma que ASP.NET *Web Pages*, mas têm a vantagem de poderem ser facilmente reutilizados.

De forma a integrar as peças, i.e. ligar os *User Controls* às páginas do MOSS, para que as interfaces estejam disponíveis para os utilizadores no MOSS, foi desenvolvida uma *Peça Web* que permite que qualquer *User Control*, desenvolvido pela equipa em ASP.NET, possa funcionar de forma perfeitamente integrada dentro do MOSS. Esta *Peça Web* possui ainda a hipótese de ser configurável para passar parâmetros para o *User Control*, o que é realizado por introdução desses parâmetros na caixa de texto dos *Parameters* da zona de *Configurations* (Figura 13).

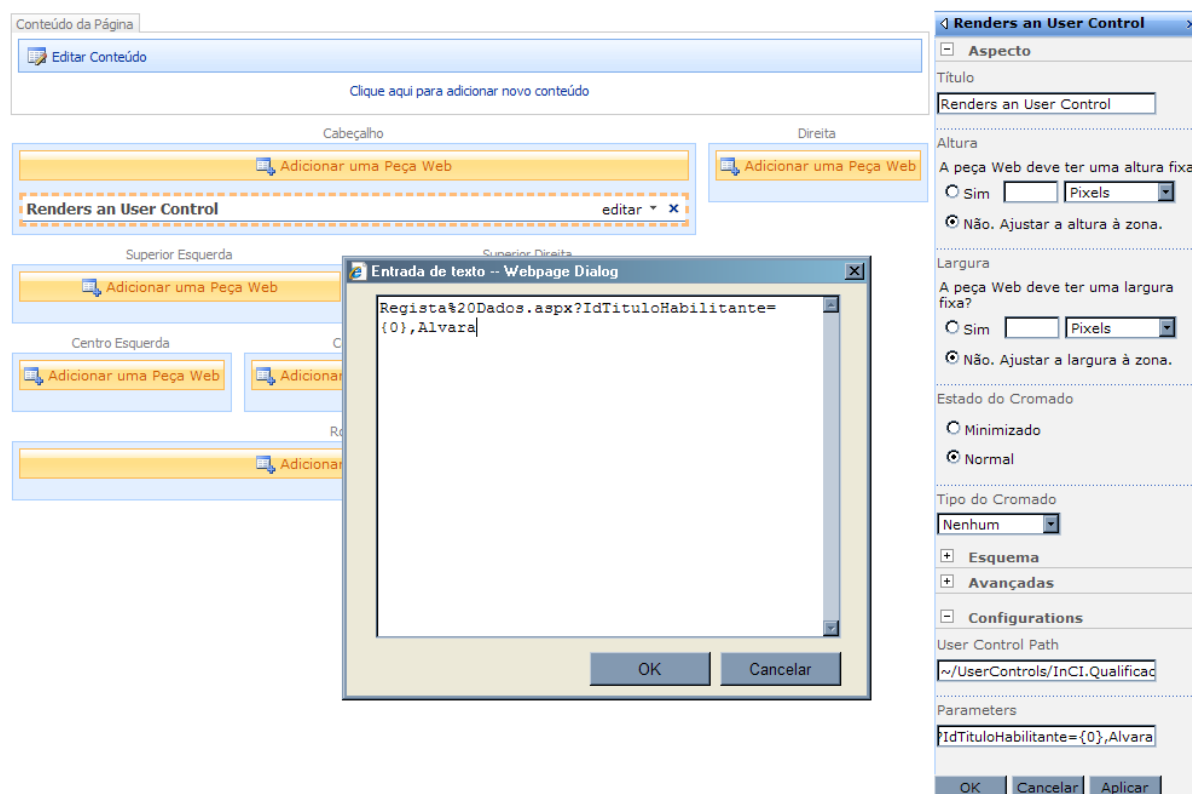


Figura 13 - Peça Web de “renderização” de *User Controls*

A utilização desta *Peça Web* confere à equipa de desenvolvimento uma grande liberdade na construção dos formulários. Está assim totalmente sob o controlo da equipa o desenvolvimento, que pode ser feito e testado localmente relativo a toda a parte da lógica de negócio, conferindo uma aceleração geral a esse desenvolvimento, ficando no fim apenas a faltar alguns testes do *user control* quando inserido dentro do ambiente MOSS.

No entanto, esta abordagem não permite que todas as funcionalidades sejam desenvolvidas apenas localmente, pois, frequentemente havia dependência com o MOSS.

Dos serviços que o aluno esteve directamente envolvido, foi possível desenvolver localmente a prova de conceito no *Windows Workflow Foundation* (ver 4.3), os serviços do processo de concessão do Departamento de Qualificação (ver 4.5) e a geração de documentos (ver 4.7) na sua parte de criação do documento, pois nos serviços de obtenção dos *templates* já estaria integrado no MOSS, onde são armazenados numa biblioteca de documentos. Todas as restantes funcionalidades e serviços desenvolvidos pelo aluno necessitaram de estar integrados directamente no MOSS, nomeadamente o processo de marcação de férias, pela necessidade de utilização das hierarquias de utilizadores que está exposta no MOSS, a Gestão Documental que tem como base uma biblioteca de documentos do *SharePoint* (ver 4.6), e o módulo de alertas (ver 4.8) que utiliza uma lista de *Sharepoint* como armazém de dados.

4.3 Prova de conceito no *Windows Workflow Foundation*

A componente de programação do presente projecto iniciou-se com a construção de uma prova de conceito (PDC) na plataforma de *Workflow Foundation* (WF). A utilização desta plataforma é um dos objectivos do projecto global da InCI, e coube ao autor deste documento

perceber quais as potencialidades da mesma, através da construção de uma PDC sobre um caso real de um processo que posteriormente irá integrar a solução.

Um dos aspectos a salientar é o facto de esta PDC servir não só para construir um exemplo, mas também para generalizar ao máximo os serviços necessários à utilização da plataforma, procurando que o tempo de programação dos restantes elementos da equipa seja o mais reduzido possível, e também a fácil integração de novos elementos, que não possuam conhecimentos sobre a plataforma. As secções seguintes apresentada de seguida os serviços *core* e classes que foram desenvolvidos nesta PDC.

4.3.1 *Workflow Helper*

O serviço de *Workflow Helper* (WH) é composto por um conjunto de funções que auxiliam na criação e manipulação dos *workflows*.

A função de inicialização de *workflows* (*Start*) tem como parâmetros de entrada o tipo de *workflow*, o *runtime* de *workflow* (que é iniciado no *glosal.aspx*), e o serviço de *Manual Workflow Scheduler Service* (MWSS).

O MWSS adquire vantagem em ser utilizado pois, permite controlar o número de *threads* que são utilizados pela solução, sendo neste caso usada a *thread* que é disponibilizada pela aplicação que o aloja (ASP.NET).

A função de *Start* como objectivo simplificar a tarefa aos programadores, obtendo o MWSS que é passado como *out* (o argumento é passado como referência) do *runtime* do *workflow*, inicializando aqui a variável, cria um novo *workflow* do tipo que recebe como parâmetro e inicializa o *workflow*. Como é utilizado o MWSS tem que se executar a função de *RunWorkflow* para que o *workflow* execute. No final da função é retornado o identificador único (GUID - Globally Unique Identifier) da instância de *workflow* que foi inicializada.

Como já foi dito anteriormente a plataforma disponibiliza um conjunto de serviços base, e disponibiliza uma função para obter esses serviços do *runtime* do *workflow*. No entanto se o serviço não tiver sido adicionado ao *runtime*, a função devolve um erro. Para “corrigir” essa situação foi implementada no WH uma função que se encarrega disso. Esta função denomina-se de *GetService* e tem como parâmetro o *runtime* do *workflow*, utiliza a função de *GetService* do *runtime* (que no caso de o serviço já estar adicionado ao *runtime* devolve-o, caso contrário devolve *null*). No caso de o serviço ainda não estar no *runtime*, esta função do WH encarrega-se de adicionar o serviço ao *runtime* e devolve o serviço.

4.3.2 *Workflow Input*

A classe de *Workflow Input* (WI) define um conjunto de propriedades que são transversais aos processos de negócio do projecto. Adicionalmente possui um *Type Parameter* que expõe acesso ao detalhe do processo, contendo os restantes dados que são inerentes e específicos de cada processo e que logicamente não são objectivados pelas restantes propriedades mais genéricas da classe.

Esta classe agrega assim todos os dados que são necessários passar para o *runtime* do *workflow*.

4.3.3 Serviço *Workflow*

O serviço de *workflow* é implementado numa classe que é chamada pelos *User Controls* nos Serviços de Apresentação. Este serviço irá ser composto por tantas funções quantas as que estejam definidas em eventos no *workflow*, i.e., o *workflow* terá um evento de arranque e vários outros até terminar, cada um desses eventos representa um momento/movimento no *workflow*.

A função que corresponde ao arranque do *workflow* é diferente das restantes, pois possui a lógica de criação do *workflow* recorrendo à função de *Start* do WH. O GUID retornado por essa função é então utilizado pelos processos e guardado dentro de estruturas de dados apropriadas, chamadas de ‘Detalhe do Processo’.

As restantes funções utilizam o GUID para dar continuidade ao processo, através da função de *RunWorkflow* do MWSS que é obtido pelo WH.

Todas as funções recebem como parâmetro o WI que é transmitido para o *workflow*.

4.3.4 Interface External Data Exchange

A interface de *External Data Exchange* (EDE) serve para definir todos os eventos que constituem um *workflow*. A seguida figura (Figura 14) apresenta de um exemplo desta interface.

```
[ExternalDataExchange]
public interface IRegistoServiceEDE
{
    /// <summary>
    /// Evento registar.
    /// </summary>
    event EventHandler<RegistoEventArgs> Registrar;
    /// <summary>
    /// Evento validar.
    /// </summary>
    event EventHandler<RegistoEventArgs> Validar;
    /// <summary>
    /// Evento consultar.
    /// </summary>
    event EventHandler<RegistoEventArgs> Consultar;
    /// <summary>
    /// Evento actualizar.
    /// </summary>
    event EventHandler<RegistoEventArgs> Actualizar;
    /// <summary>
    /// Evento processamento.
    /// </summary>
    event EventHandler<RegistoEventArgs> Processamento;
    /// <summary>
    /// Evento terminar.
    /// </summary>
    event EventHandler<RegistoEventArgs> Terminar;
```

Figura 14 - Exemplo de interface de *External Data Exchange*

A seguinte figura (Figura 15), captada da ferramenta de desenvolvimento (Visual Studio), ilustra a utilização desta interface na definição de duas propriedades da actividade de *HandleExternalEvent*: *InterfaceType* e *EventName*. O *InterfaceType* define o tipo de interface

que se quer utilizar na actividade, enquanto o *EventName* indica o evento que é associado à actividade.

Para a definição do *InterfaceType* e conforme confere a Figura 14, a interface tem de ser marcada com a propriedade *ExternalDataExchange*, e os eventos serão posteriormente listados na janela da propriedade *EventName*.

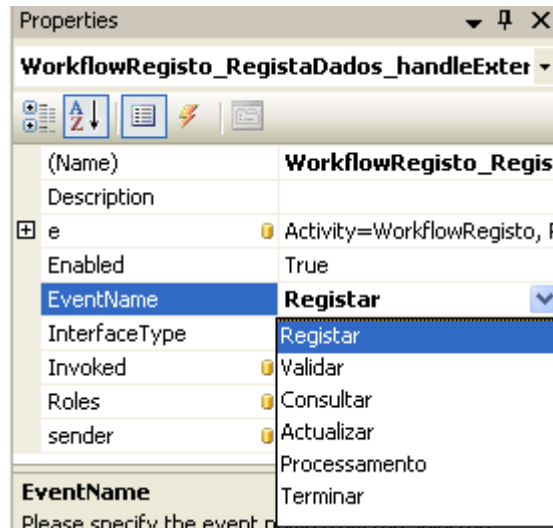


Figura 15 - Propriedades de Actividade de *Handle External Event* obtidas da interface de *External Data Exchange*

Adicionalmente à interface, é tipicamente colocado no mesmo ficheiro onde se define a classe que implementa a interface, uma classe que implementa o *ExternalDataEventArgs*, que representa os dados que são enviados quando um evento ocorre numa actividade de *HandleExternalEventActivity*.

4.3.5 Serviço que implementa interface de EDE

O serviço que implementa a interface de EDE deve conter a lógica associada aos eventos aí declarados. Este serviço permite fazer a ligação entre os dados passados, os eventos e o próprio *workflow* por acção da chamada aos eventos aliado à construção de um parâmetro específico que é aí transmitido (*ExternalDataEventArgs*).

4.3.6 Desenho do *Workflow*

O desenho do *workflow* é uma tarefa que está simplificada pelo Visual Studio, que possui um *designer* próprio para o efeito. Os tipos de *workflows* adoptados pelo projecto são máquinas de estado, como tal podem ser facilmente adicionados novos estados ao *workflow*.

Cada novo estado solicita que seja introduzida uma actividade de *EventDrivenActivity*, que no fundo representa uma actividade cuja execução é inicializada por um evento. A Figura 16 apresenta um estado com uma dessas actividades inseridas dentro do estado.

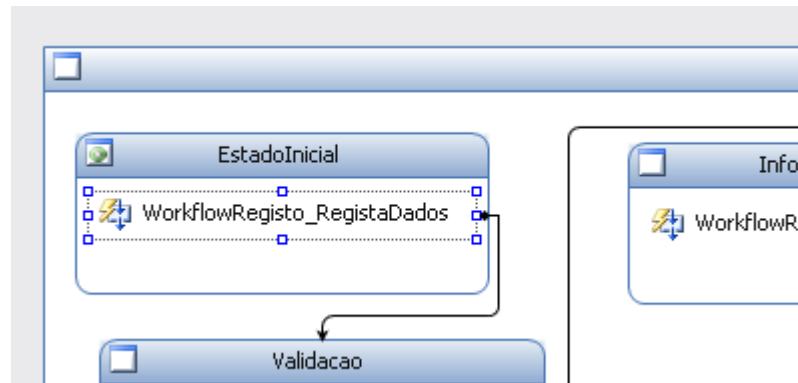


Figura 16 - Estado *workflow* com actividade de *EventDriven*

Um estado pode ter várias destas actividades o que possibilita várias acções no mesmo estado com diversos resultados finais para o estado.

Quando se está dentro dessa actividade, o primeiro passo a ser definido tem de ser uma actividade que suporte a interface *IEventActivity*. As duas actividades que se qualificam para isso são o *HandleExternalEvent* e o *Delay* [5].

A PDC também define que se deve ter em consideração uma actividade de *TransactionScope* de modo a garantir a transacção total do evento. A partir daí surge a definição das restantes actividades específicas do processo, estado ou evento.

4.3.7 Call Workflow Activity

A PDC previu ainda que fossem feitos alguns teste no que reporta à chamada de *workflows* externos, i.e., um *workflow* ter internamente, num dos seus passos, a chamada à execução de um outro *workflow*. Esta actividade é transversal a várias áreas e pode ser reutilizado, centralizando assim uma possível mudança num único ponto.

A plataforma do WF contempla a invocação de *workflows* de forma assíncrona, ou seja, um dado *workflow* tem uma actividade que chama outro *workflow* que funciona de forma perfeitamente independente do que lhe deu origem. No entanto, foi desde logo, no período de análise, identificada a necessidade de invocar outros *workflows* e que fossem executados de forma síncrona, podendo mesmo a devolver um resultado ao “criador”. Tendo isto em consideração foi desenvolvida esta actividade que contempla esta situação.

4.3.8 Aplicação em caso real

A prova de conceito foi concretizada com a aplicação dos serviços desenvolvidos num caso real. O processo que foi utilizado está representado na Figura 17, que foi obtida do documento de análise do projecto, e caracteriza o processo de *Help-desk* da Direcção Administrativa, Financeira e de Recursos Humanos.

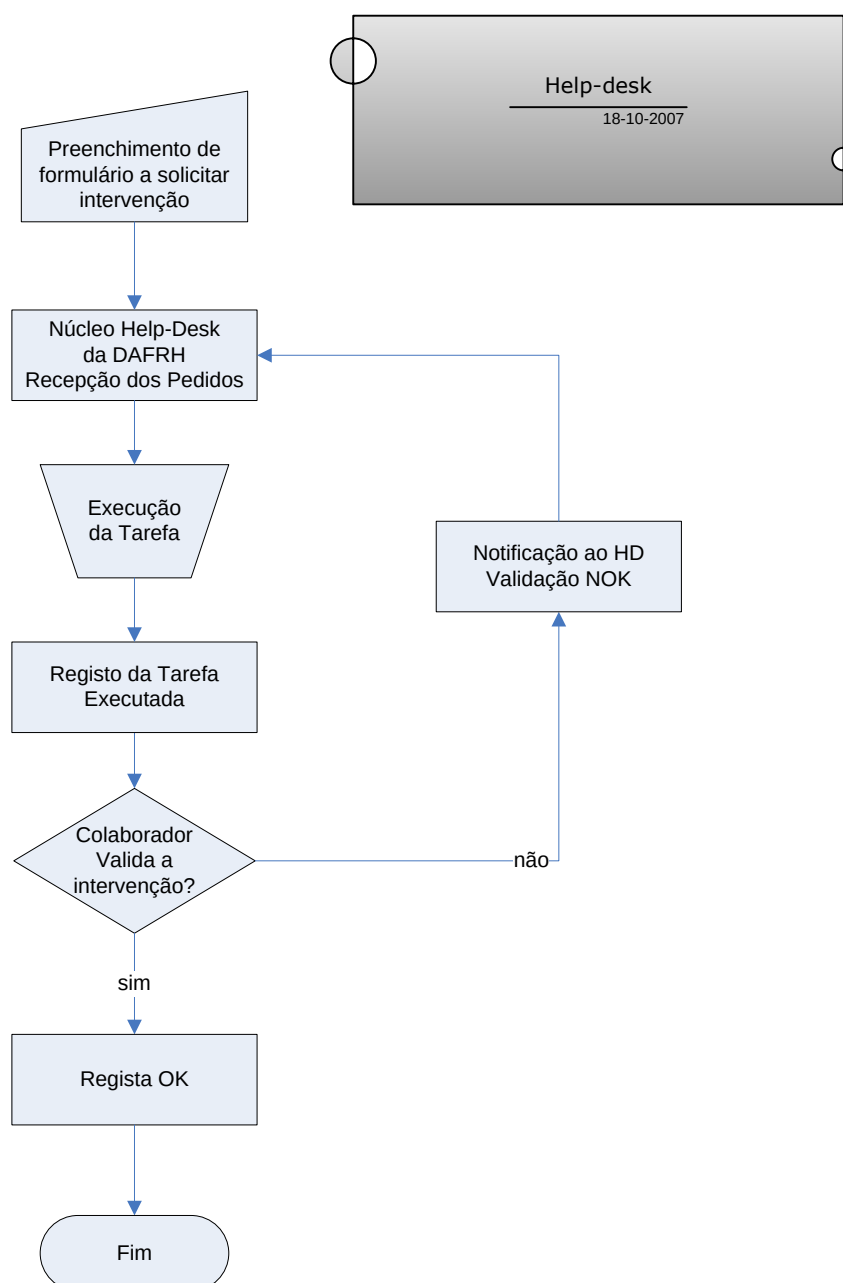
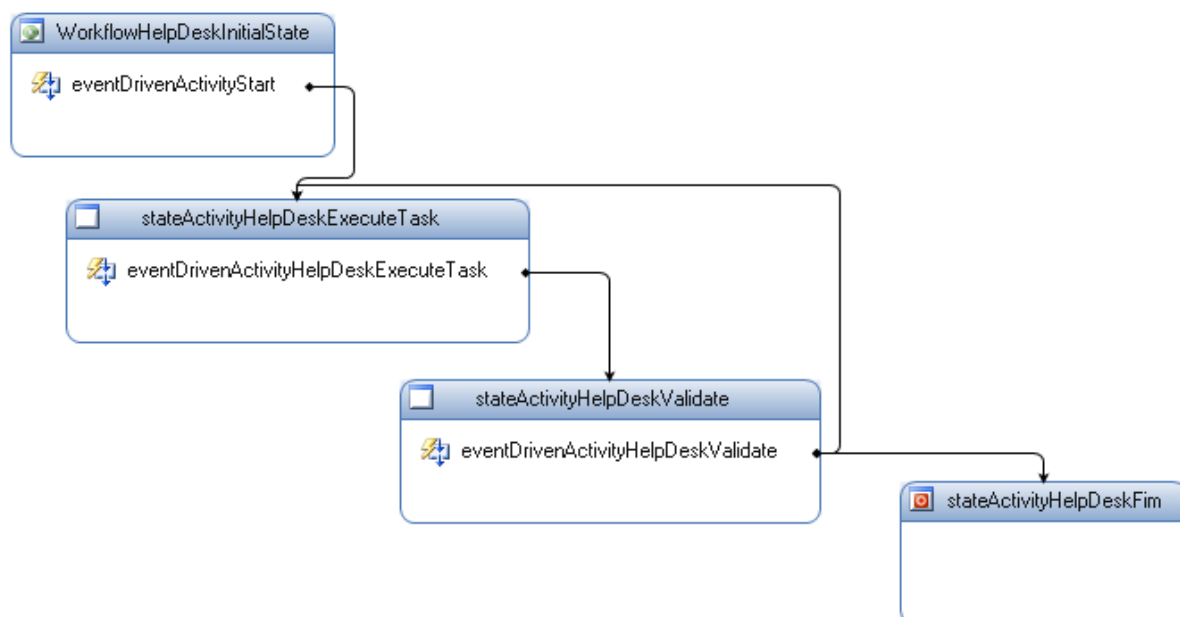


Figura 17 - Processo de *Help-Desk*

Neste processo foram identificados um conjunto de estados, que são traduzidos num *workflow* de máquina de estado apresentado na Figura 18.

Figura 18 - *Workflow Help-desk*

Este *workflow* tem quatro estados: Inicial, Executar tarefa, Validação e Final.

O processo é iniciado com a criação de uma nova tarefa de *help-desk*, que inicializa o *workflow*. O *workflow* evolui para o estado de *stateActivityHelpDeskExecuteTask* (Executar tarefa), onde o utilizador pode visualizar a tarefa que tem de executar. Após ter terminado, envia resultado da tarefa para aprovação, fazendo avançar o *workflow* para o estado de *stateActivityHelpDeskValidate* (Validação). Os utilizadores, com perfil de validação, podem aceder ao detalhe da tarefa para procederem à validação, levando o *workflow* a seguir um de dois caminhos, no caso da tarefa ser validada o *workflow* passa para o estado *stateActivityHelpDeskFim* (Final), se a tarefa não for assinalada como válida volta novamente para o estado de *stateActivityHelpDeskExecuteTask*, formando um ciclo até que a tarefa executada seja assinalada com válida.

4.4 Processo de Marcação de Férias do DAFRH

O processo de marcação de férias gerido pela Direcção Administrativa, Financeira e de Recursos Humanos (DAFRH) consiste em fornecer aos colaboradores do InCI uma funcionalidade para poderem marcar as suas férias pessoais. Este processo obedece a um ciclo de aprovação

A interface gráfica do processo de marcação de férias do DAFRH foi desenvolvida com o intuito de ser amigável e intuitiva para qualquer utilizador deste módulo. Após indicar o ano referente aos dias de férias que se pretende (valores que são obtidos de uma fonte de dados externa - aplicação de Recursos Humanos), é apresentado ao utilizador um calendário onde este pode marcar os dias de férias que o utilizador pretende (Figura 19).

Férias e Faltas referentes ao ano de 2008

Gravar Alterações Gravar e Pedir Aprovação | Mapa Geral

◀ Ano: 2008 ▶ Hoje

Janeiro de 2008							Fevereiro de 2008							Março de 2008							Abril de 2008						
seg	ter	qua	qui	sex	sáb	dom	seg	ter	qua	qui	sex	sáb	dom	seg	ter	qua	qui	sex	sáb	dom	seg	ter	qua	qui	sex	sáb	dom
31	F	2	3	4	5	6	28	29	30	31	1	2	3	25	26	27	28	29	1	2	31	1	2	3	4	5	6
7	8	9	10	11	12	13	4	E	6	7	8	9	10	3	4	5	6	7	8	9	7	8	9	10	11	12	13
14	15	16	17	18	19	20	11	12	13	14	15	16	17	10	11	12	13	14	15	16	14	15	16	17	18	19	20
21	22	23	24	25	26	27	18	19	20	21	22	23	24	17	18	19	20	F	22	P	21	22	23	24	F	26	27
28	29	30	31	1	2	3	25	26	27	28	29	1	2	24	25	26	27	28	29	30	28	29	30	1	2	3	4
4	5	6	7	8	9	10	3	4	5	6	7	8	9	31	1	2	3	4	5	6	5	6	7	8	9	10	11

Maio de 2008							Junho de 2008							Julho de 2008							Agosto de 2008						
seg	ter	qua	qui	sex	sáb	dom	seg	ter	qua	qui	sex	sáb	dom	seg	ter	qua	qui	sex	sáb	dom	seg	ter	qua	qui	sex	sáb	dom
28	29	30	F	2	3	4	26	27	28	29	30	31	1	30	1	2	3	4	5	6	28	29	30	31	1	2	3
5	6	7	8	9	10	11	2	3	4	5	6	7	8	7	8	9	10	11	12	13	4	5	6	7	8	9	10
12	13	14	15	16	17	18	9	F	11	12	F	14	15	14	15	16	17	18	19	20	11	12	13	14	F	16	17
19	20	21	F	23	24	25	16	17	18	19	20	21	22	21	22	23	24	25	26	27	18	19	20	21	22	23	24
26	27	28	29	30	31	1	23	F	25	26	27	28	29	28	29	30	31	1	2	3	25	26	27	28	29	30	31
2	3	4	5	6	7	8	30	1	2	3	4	5	6	4	5	6	7	8	9	10	1	2	3	4	5	6	7

Setembro de 2008							Outubro de 2008							Novembro de 2008							Dezembro de 2008						
seg	ter	qua	qui	sex	sáb	dom	seg	ter	qua	qui	sex	sáb	dom	seg	ter	qua	qui	sex	sáb	dom	seg	ter	qua	qui	sex	sáb	dom
25	26	27	28	29	30	31	29	30	1	2	3	4	F	27	28	29	30	31	F	2	24	25	26	27	28	29	30
1	2	3	4	5	6	7	6	7	8	9	10	11	12	3	4	5	6	7	8	9	F	2	3	4	5	6	7
8	9	10	11	12	13	14	13	14	15	16	17	18	19	10	11	12	13	14	15	16	F	9	10	11	12	13	14
15	16	17	18	19	20	21	20	21	22	23	24	25	26	17	18	19	20	21	22	23	15	16	17	18	19	20	21
22	23	24	25	26	27	28	27	28	29	30	31	1	2	24	25	26	27	28	29	30	22	23	24	F	26	27	28
29	30	1	2	3	4	5	3	4	5	6	7	8	9	1	2	3	4	5	6	7	29	30	31	1	2	3	4

22 dias de férias marcados. 0 dias disponíveis de 22 dias de férias possíveis.

Figura 19 - UI de Marcação de Férias

O utilizador pode apenas gravar os dias de férias para organização pessoal, ou poderá também enviar as férias para aprovação. Esta última acção inicializa o *workflow* de Marcação de Férias. Este pedido de aprovação é encaminhado para o superior hierárquico de acordo com a estrutura definida pelo InCI que é gerida no *Active Directory* (AD) e no MOSS.

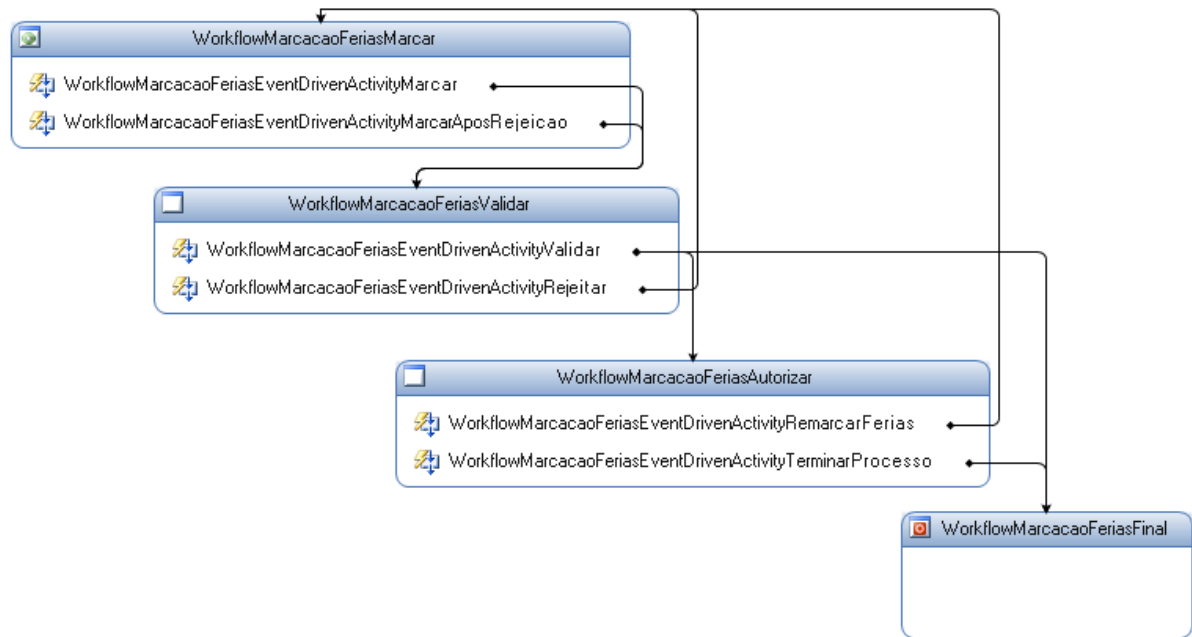


Figura 20 - *Workflow* de Marcação de Férias

Os utilizadores que têm subordinados, quando recebem pedidos de marcação de férias podem rejeitar ou aprovar, através de acesso a uma listagem desses utilizadores, onde se pode seleccionar cada um para aceder ao detalhe das férias, o que corresponde ao *User Control* apresentando anteriormente, mas em modo de leitura. A acção de aprovação corresponde a um evento que despoleta novamente o *workflow*.

Se a marcação de férias for rejeitada o utilizador superior deverá indicar o motivo da rejeição de modo a que o subordinado saiba o que tem de corrigir para pedir nova aprovação.

Se as férias forem aprovadas, o processo passa para um estado de aceitação temporária (*WorkflowMarcacaoFeriasAutorizar*), isto porque a lógica de marcação de férias do InCI indica que se existirem mais níveis de hierarquia o utilizador superior poderá rejeitar um utilizador que tenha subordinados, o que pode levar a que este possa propagar a rejeição para “baixo” de modo a poder, por exemplo, conciliar as suas férias com um dos seus subordinados.

Adicionalmente a isto, um utilizador que tenha subordinados só poderá pedir aprovação das suas férias quando todos os utilizadores subordinados tiverem as suas férias marcadas e aprovadas. Esta ideologia, com reutilização do processo, funciona até a um nível onde o superior não tenha nenhum utilizador acima dele, o que significa que poderá, no caso de aprovar todos os subordinados, propagar a acção de aprovação final de todos os utilizadores que estão colocados na árvore hierárquica.

A lógica do serviço faz também validações aos dados introduzidos, verificando, por exemplo, se o pedido do utilizador não excede o número de dias de férias que tem atribuído, ou se, pelo contrário, fica com dias de férias por marcar.

Serviços de Camada de acesso a dados

Os serviços de camada de acesso a dados foram excepcionalmente obtidos através de *Typed DataSets* que foram construídos no Visual Studio.³

Base de dados

A camada de dados para suporte a este processo comporta uma única tabela onde são armazenados os dias de férias marcados por cada utilizador. Cada dia de férias que seja marcado nos calendários corresponde a um novo registo na tabela de Férias.

4.5 Processo de Concessão do Departamento de Qualificação

O processo de concessão do Departamento de Qualificação (DQ) consiste na criação de empresas, caso ainda não existam, para atribuir um alvará, título de registo, licença de mediação ou licença de angariação a essa empresa. São recolhidos vários dados associados à empresa, como por exemplo, quadro pessoal, quadro técnico, representante legal ou documentos.

O processo de concessão do DQ necessita de validações a algumas capacidades e quadros que as empresas que se candidatam à concessão devem possuir. Estas validações actuam sobre os dados que são recolhidos.

As validações a considerar são validações de condições iniciais – que ocorrem no momento em que o pedido de concessão de uma empresa acontece (e.g. pedido de alvará de construção).

Todos estes serviços foram implementados pelo aluno, estando agregados numa classe que se situa na camada de Serviços de Negócio. Os processos que necessitam desta lógica inserida nos *workflows* acedem a essa camada passando por parâmetro um conjunto de dados indispensáveis a estas validações.

Os serviços descritos nos parágrafos seguintes implementam essas validações.⁴

Capacidade Técnica

A validação da capacidade técnica envolve duas verificações: Quando Técnico e Quadro Pessoal apresentados a seguir.

Quadro Técnico

³ A ideologia de desenvolvimento do InCI estipulada pelo gestor de projecto indica que tudo o que fosse desenvolvido e que estivesse a funcionar correctamente sem grande diminuição de desempenho deveria ser mantido, sendo que qualquer evolução ao modo de trabalho será apenas considerado desse momento para a frente. Isto porque no final de obtenção deste processo já estava a vigorar uma nova alternativa à construção da camada de acesso a dados dos Serviços de Dados.

⁴ Os serviços que foram desenvolvidos são apresentados de forma descontextualizada para não comprometerem a lógica que constitui o processo de concessão do InCI, considerando suficiente para demonstrar os vários desafios a que o aluno foi sujeito.

Uma empresa quando inicia um processo de concessão especifica a lista de classes (habilitações de construção da empresa) a que se candidata e quais as pessoas que compõem os quadros da empresa. Estas listagens são utilizadas para verificar se a empresa por um lado possui o número mínimo de pessoas (e.g. engenheiros técnicos) no quadro para executar as habilitações de classe a que se candidata e também para verificar se tem nos seus quadros pessoas com qualificação para essas habilitações. O resultado destas operações é uma lista de excepções, no caso de não cumprir alguma das regras, lista essa que é será vazia se a empresa possui o quadro necessário.

Quadro Pessoal

A verificação do Quadro Pessoal incide sobre o número de encarregados e operários de grupos que têm de ser em número suficiente para a classe da empresa. O resultado funciona de forma semelhante ao anterior.

Capacidade económico-financeira

A capacidade económico-financeira verifica se a empresa, de acordo com o regime em que está (probatório = inicial, ou normal), tem um conjunto de valores de acordo com um quadro que é estipulado pelo InCI.

Capacidade profissional

A capacidade profissional avalia se as pessoas, que conferem esse valor às empresas, possuem na sua formação o grau necessário para executar a actividade.

Situação Regularizada

A situação regularizada avalia se uma empresa tem um conjunto de valores normalizados, i.e. não tem coimas por pagar, nem dívidas à segurança social, e finalmente, não tem dívidas às finanças, estas verificações fazem integração com as respectivas entidades competentes, sendo que no caso de não estarem operacionais a base de dados do projecto contempla uma cópia de segurança desses valores (sendo datados a um determinado momento e servindo apenas de recurso).

4.6 Gestão Documental

A gestão documental é uma funcionalidade (*feature*) que foi desenvolvida para o projecto do InCI, tendo como objectivo ser a biblioteca de toda a documentação que os processos internos e externos possam necessitar. Tratando-se de uma funcionalidade central a muitas outras funcionalidades deve satisfazer as necessidades de todas elas. Para tal foram enumerados um conjunto de atributos que definiam os documentos, tipificando-os em 3 categorias:

- Documentos Internos – para processos internos do InCI, i.e. entre departamentos do InCI sem afectação a entidades externas
- Documentos Entrada – que chegam ao expediente através de várias origens (CTT, *Upload* no Portal, Fax, ou *Email*).
- Documentos Saída – que foram tratados pelo expediente

Os atributos destes documentos podem ser visualizados no Anexo B.

A construção de uma *feature* de *Sharepoint* é alcançada através da criação de vários ficheiros *XML* e um *C#* que são descritos de seguida:

- *elementManifest.xml* – define os ficheiros que fazem parte da *feature* (e.g. ficheiros de imagens)
- *Feature.xml* – contém as definições da *feature*, tais como Id, Nome, Descrição etc.

```
<?xml version="1.0" encoding="utf-8" ?>
<Feature xmlns="http://schemas.microsoft.com/sharepoint/"
  Id="{5D68F161-2B0A-4e24-BF99-EEA0C8232E80}"
  Title="INCI GD"
  Description="INCI Gestao Documental"
  Scope="Site"
  ReceiverAssembly="INCI.IntranetWeb, Version=1.0.0.0, Culture=neutral, PublicKeyToken=5d44aa5aad34155"
  ReceiverClass="INCI.Sharepoint.GestaoDocumental.GestaoDocumentalFeatureInstaller"
  Hidden="FALSE">
  <ActivationDependencies>
    <ActivationDependency FeatureId="4789819E-A21A-44c0-9BF1-C918AC089467"/>
  </ActivationDependencies>
  <ElementManifests>
    <ElementManifest Location="SiteColumn.xml" />
    <ElementManifest Location="ContentTypesDocuments.xml" />
  </ElementManifests>
</Feature>
```

Figura 21 – Exemplo de XML de *feature* de *SharePoint*

- *SiteColumn.xml* – Contém as definições de colunas do site, para serem utilizadas pelos tipos de conteúdos referentes aos 3 tipos de documentos apresentados.

```
<Field ID="{6E965FD0-F6E0-442a-A9C9-24E17E4F2A53}"
  Name="INCIGDOrigemSaida"
  DisplayName="Origem"
  Type="Choice"
  Group="INCI"
  FillInChoice="FALSE"
  Required="FALSE">
  <CHOICES>
    <CHOICE>Fax</CHOICE>
    <CHOICE>Email</CHOICE>
    <CHOICE>CTT</CHOICE>
  </CHOICES>
</Field>

<Field ID="{DC2CF397-D821-490a-9BAE-5F35A7D2886B}"
  Name="INCIGDConfidencial"
  DisplayName="Confidencial"
  Group="INCI"
  Type="Boolean"
  Required="FALSE"
/>

<Field ID="{F8380D06-F607-461a-BF4D-1CE0069AD4E7}"
  Name="INCIGDObservacoes"
  DisplayName="Observações"
  Group="INCI"
  Type="Text"
  Required="FALSE"
/>
```

Figura 22 – Exemplo de XML de colunas

- *ContentTypesDocuments.xml* - Propriedades transversais foram colocadas num tipo de conteúdo que é chamado de documento base e do qual herdam todos os outros tipos de documento.
- *GestaoDocumentalFeatureInstaller.cs* – define um conjunto de funções onde se podem executar acções/código em determinados momentos em que a *feature* executa, nomeadamente instalação, desinstalação, activação e desactivação.

No momento da instalação da *feature* são criadas várias bibliotecas de documentos que servem de suporte ao negócio do InCI.

- Documentos – biblioteca principal e central dos documentos. É a biblioteca que aloja os documentos do expediente (entrada e saída), internos,
- *Templates* Do InCI – biblioteca de documentos que aloja os *templates* que servem de base para a geração automática de documentos (ver detalhe em 4.7).
- Biblioteca Temporária – trata-se da biblioteca que aloja os documentos que estão num estado temporário, como por exemplo no meio de um processo, que tem um conjunto de documentação que é gerada (cartas de notificações) que só enviará esses documentos para o expediente no final do processo/*workflow*.

Gestão Documental Application Programming Interface

A Gestão Documental *Application Programming Interface* (GDAPI) é uma classe central neste serviço de Gestão Documental, e contém um conjunto variado de funções que auxiliam na criação e manutenção dos documentos das bibliotecas anteriormente mencionadas. As principais funções que constituem a classe são as seguintes:

- CarregarDocumento – Esta função faz referência a várias funções que auxiliam no carregamento de documentos para as bibliotecas de documentos. Sendo que existem opções para as seguintes vertentes:
 - Internet – documentos vindos da Internet;
 - Interno;
 - Entrada;
 - Saída.

Os carregamentos diferenciam-se entre si por atribuir, por omissão, aquando da chamada à função, propriedades específicas que os identificam como sendo de um determinado tipo.

- ActualizarDocumento – funções de actualização de documento.
 - IdProcesso – quando no âmbito de um processo é carregado um documento através das funções acima apresentadas, e se estivermos a falar do início do processo, o identificador do processo ainda não está disponível pelo que é necessário criar lógica de actualização de propriedades do documento no momento certo. De modo a auxiliar nessa tarefa esta função foi desenhada para actualizar esse Id.
 - AtributosDeDocumento – esta função recebe um objecto com todos os atributos que são possíveis de existir num documento, que são utilizados para actualizar as propriedades do documento.
 - ActualizarDocumento – referenciando um determinado id de documento, é também possível actualizar os *bytes* do documento, i.e. actualizar o próprio ficheiro.
- Mover documentos entre bibliotecas (incluindo todos os “meta-dados” que têm)
- Pesquisa de documentos

A pesquisa de documentos é conseguida através da construção de uma *query* na linguagem CAML⁵ (Collaborative Application Markup Language).

Business Data Catalog

A funcionalidade de *Business Data Catalog* (BDC) do MOSS fornece uma maneira fácil de integrar dados de negócio de aplicações de um servidor de *back-end*, tais como SAP ou Siebel, com o portal colaborativo de modo a fornecer soluções ricas para os utilizadores finais sem escrever nenhum código. [6]

Os tipos de conteúdos que foram anteriormente indicados têm necessidade de estabelecer ligação à base de dados em algumas das suas colunas (e.g. propriedade de tipo de documento), isto pode ser alcançado através do BDC.

O BDC é importado de um ficheiro XML que foi criado contendo todas as entidades necessárias ao projecto. A Figura 22 apresenta um exemplo de um ficheiro deste tipo.

```
<Entity EstimatedInstanceCount="10000" Name="TiposDocumento">
  <Identifiers>
    <Identifier TypeName="System.Int32" Name="IdTipoDocumento" />
  </Identifiers>
  <Methods>
    <Method Name="Find_TiposDocumento">...
    <Method Name="FindAll_TiposDocumento">...
    <Method Name="Find_TiposDocumento_By_Name">
      <Properties>
        <Property Name="RdbCommandType" Type="System.Data.CommandType, System.Data, Version=2.0.0.0, Culture=neutral, PublicKey
        <Property Name="RdbCommandText" Type="System.String">
          Select "IdTipoDocumento","Nome","Sigla" from TiposDocumento
          where Nome LIKE '%'+@Nome+'%' AND (Sigla LIKE '%'+@Sigla+'%' OR Sigla IS NULL)
        </Property>
      </Properties>
      <FilterDescriptors>
        <FilterDescriptor Type="Comparison" Name="Nome" />
        <FilterDescriptor Type="Comparison" Name="Sigla" />
      </FilterDescriptors>
      <Parameters>
        <Parameter Direction="In" Name="@Nome">...
        <Parameter Direction="In" Name="@Sigla">...
        <Parameter Direction="Return" Name="@TiposDocumento">...
      </Parameters>
      <MethodInstances>
        <MethodInstance Type="Finder" ReturnParameterName="@TiposDocumento" ReturnTypeDescriptorName="TiposDocumentoDataReader"
      </MethodInstances>
    </Method>
  </Methods>
</Entity>
```

Figura 23 - Exemplo de XML de *business data catalog*

O ficheiro define métodos para aceder aos dados. A figura mostra em maior detalhe o método para pesquisa de tipos de documento, que utiliza SQL com filtragem através de dois parâmetros de entrada (*Direction='In'*). Existem vários tipos de métodos, mas apenas os seguintes foram utilizados no projecto:

- *Finder* – devolve instâncias da entidade;
- *SpecificFinder* – devolve exactamente uma instância (utilizado por exemplo para visualizar o detalhe de uma entidade);
- *IdEnumerator* – devolve uma lista de IDs (chaves únicas para cada entidade).

⁵ O CAML (*Collaborative Application Markup Language*) é uma linguagem baseada em XML (eXtensible Markup Language) utilizada pela Microsoft em tecnologias SharePoint [14].

4.7 Geração automática de documentos

Ao longo dos processos que constituem o projecto existem casos em que é necessário a geração automática de documentos, assinados digitalmente, com base em *templates*.

A estrutura de criação de um documento foi analisada de forma a poder contemplar facilmente novos *templates*. Existe uma biblioteca de documentos, no site de Gestão Documental do portal de intranet do InCI, específica para a manutenção de *templates* do InCI.

O utilizador quando realiza o *upload* de um novo documento para a biblioteca tem de introduzir um conjunto de propriedades para complementar a informação do *template*. Estas propriedades são utilizadas para obter dados da base de dados do InCI. A seguinte figura apresenta essas propriedades:

TemplatesDoInCI: Baixa Alvará

✕ Eliminar Item |
 ABC A verificar a ortografia...
* indica um campo obrigatório

IdTipoTemplate *	1
Nome *	Baixa Alvará .docx
Título	Notificação de Baixa de Álvára
Coluna Identificador	IdTipoProcesso
Colunas	Descritivo
Data de início *	01-03-2008
Data de validade	31-03-2008
Tabela/View	TiposProcesso
IdTipoDocumento	10
IdTipoProcesso	22

Criado em 31-03-2008 17:19 por Conta de Sistema
 Última modificação em 31-03-2008 17:19 por Conta de Sistema

Figura 24 - Propriedades dos *templates*

- IdTipoTemplate – identificador do tipo;
- Nome – do ficheiro;
- Título – texto com o nome;
- Tabela/View – a tabela ou *view* de onde se lêem as colunas acima identificadas;
- Colunas – as colunas que se querem ler da tabela/*view* da base de dados;
- Coluna Identificador – a coluna que identifica o registo como único na base de dados;
- Data de início – data a partir da qual o *template* é válido;
- Data de validade – data em que o *template* deixa de vigorar;
- IdTipoDocumento – identificador do tipo de documento (ligado à base de dados por *Business Data Catalog*);
- IdTipoProcesso – o identificador do tipo de processo ao qual se aplica este *template*.

A leitura de dados é assim efectuada de forma dinâmica, bastando alterar uma das propriedades mencionadas para alterar o *template* produzido. Esta abordagem origina que para cada tipo de *template* possa ter várias entradas na biblioteca, diferenciadas pela data.

Para diferenciar as entradas relativas ao mesmo tipo de *template*, não foi utilizado o controlo de versões de ficheiros do MOSS, mas sim as propriedades de datas. Esta opção é justificada pelo facto de que quando se pretende ter uma nova versão do *template* do documento o utilizador pode submeter o *template* para o MOSS quando a data dessa versão chega, o que se torna inoportuno. Não quer isto dizer que o sistema de controlo de versões não exista, porque se o ficheiro contiver algum erro poderá ser facilmente substituído, sem que para isso seja necessário apagar totalmente o registo anterior.

Para melhor explicar o dinamismo dos *templates*, considere-se a situação em que o utilizador decide que o *template* (docx) deverá ter mais uma propriedade ou controlo. Este poderá editar as propriedades do documento no MOSS e acrescentar a nova coluna a ser lida da base de dados. O documento (docx) deverá também ser editado para reflectir esta alteração.

Quanto à geração do documento propriamente dita, é disponibilizada uma função que recebe:

- O identificador do tipo de *template* e a data que o identificam univocamente (o sistema pesquisa por *templates* que tenham um intervalo que envolva essa data);
- Identificador do registo a ser lido da base de dados;
- Lista de campo – para o caso de alguns dos dados não estarem na base de dados, mas existir em negócio.

Os dois primeiros parâmetros são utilizados para obter o *template* da gestão documental recorrendo à API do MOSS. Os bytes que são aí retornados são convertidos para um objecto *Package*⁶.

À lista de campos de campos obtidos de negócio junta-se os que são lidos da base de dados, formando um dicionário com todos os campos necessários ao *template*. Com base nesta lista de campos é criado um ficheiro XML que passará a integrar o *package* (CustomXML). O conteúdo deste documento é criado dinamicamente realizando uma navegação pelos vários nós da estrutura em árvore que constitui a lista. Este XML faz o mapeamento com os controlos que são criados no documento Word que serve de *template*. Os controlos podem ser organizados dentro de grupos.

O documento que é gerado vai ler os valores que têm de ser inseridos nos controlos a esse XML.

Um dos requisitos que o InCI tinha para esta funcionalidade era a necessidade de existir partes do documento que são repetidas, tais como, listagens de motivos de recusa, ou listagem de sub-empregados até um determinado nível. Este requisito está contemplado na solução desenvolvida, sendo que na lista de objectos que forma a árvore, poderão ser repetidos grupos o que leva na construção do documento final a que os respectivos controlos sejam também repetidos.

⁶ O *package* é uma classe abstracta que pode ser utilizada para organizar objectos numa entidade única. Um ficheiro ZIP é o formato primário, mas existem mais implementações tais como documentos XML, base de dados, ou *Web service*.

Outro requisito que também foi contemplado, foi o de poder “esconder” um determinado grupo (incluindo conteúdos e controlos), podendo assim reutilizar o mesmo *template* para várias situações.

Finalmente, todas as partes do *Package* são assinadas digitalmente, utilizando um certificado que tem de estar instalado no servidor onde a aplicação está a executar.

O resultado final é um documento assinado com os campos preenchidos.

O processo de geração de documentos contempla ainda uma situação final que actua sobre tudo o que foi acima descrito. O InCI tem de frequentemente criar notificações que são enviadas aos clientes/utilizadores da instituição, essas notificações são criadas contendo uma parte que é comum a todos os *templates* e consequentemente aos documentos, o destinatário da notificação. Esta notificação será criada e inserida na biblioteca de documentos com atributos que a identificam como sendo um documento a ser tratado pelo expediente. O expediente é encarregado de encaminhar o resto do processo.

A seguinte imagem apresenta o *user control* que foi criado para este efeito, e que é utilizado transversalmente por vários processos.

Id	Tipo	Nome	Endereço	Código Postal	Localidade	País	Email
0	Empresa	Ezequiel Valadas	Rua pires de sousa	4200-022	Paranhos	Portugal	

Figura 25 - Manutenção de destinatários de ofício

Este controlo permite que sejam adicionados mais destinatários da notificação, o que leva à geração de diversos documentos, ou que seja alterada a morada de destino de qualquer dos destinatários, i.e. das moradas que o destinatário tem registado pode seleccionar qualquer uma delas, isto porque por omissão o destinatário tem uma morada (e.g. sede social de uma empresa). Em cada uma das linhas da tabela, existe parte mais à direita um botão onde o utilizador pode pré-visualizar o documento para esse destinatário, verificando em directo as alterações que for efectuando.

4.8 Alertas

Os alertas têm de ser criados durante a execução de determinadas tarefas dos processos. O objectivo é notificar os utilizadores através de mensagens. Os alertas caracterizam-se pelas seguintes propriedades:

- Título,
- Descrição;

- Hiperligação - que é opcional e serve para o utilizador seguir de modo a encontrar mais facilmente por exemplo o processo que originou o alerta;
- Tipo – pode ser do tipo notificação ou tarefa;
- Correio electrónico – *true* caso o alerta também seja para ser enviado por correio electrónico, *false* caso contrário
- Lido – se *true* alerta foi lido, *false* caso contrário
- Notificado – *true* caso o utilizador já tenha sido notificado do alerta, *false* caso contrário

Os alertas são criados através da API do MOSS, e guardados numa lista de *Sharepoint*.

Os alertas podem ser enviados para um utilizador ou para um grupo de *Sharepoint*. Este grupo pode conter utilizador e/ou grupos da AD. Caso esses grupos existam é necessário estabelecer a devida comunicação com a AD, de modo a obter todos os utilizadores que pertençam a esse grupo.

A imagem seguinte mostra a notificação que o utilizador visualiza, no canto inferior direito do ecrã, quando entra na página inicial da intranet do InCI.

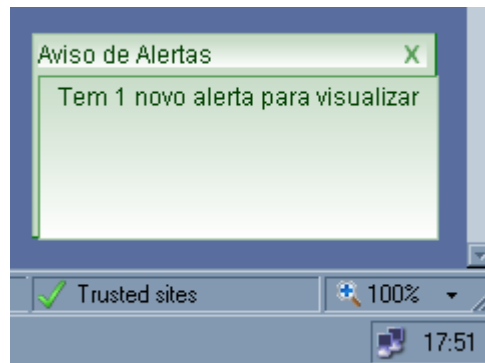


Figura 26 - Notificação de novas alertas

A notificação tem no texto uma hiperligação com a lista de alertas pessoais que foi desenvolvida à medida.

Selecione: Não Lidos Lidos Todos Nenhum				
Acções: Eliminar Marcar como lida Marcar como não lida				
Select	Título	Criado Em	Tipo	
☐	Alerta 2	01-04-2008 17:53:57		
☐	Alerta 1	01-04-2008 17:50:31		
Detalhe do Alerta Título: Alerta 1 Tipo: ☒ Notificacao Descrição: Texto do Alerta 1 Criada em: 01-04-2008 17:50:31				

Figura 27 - Listagem de alertas pessoais

Este controlo tem diversas opções de selecção (não lidos, lidos, todos e nenhum), e também acções para eliminar, marcar como lida e marcar como não lida. Estas funcionalidades foram desenvolvidas para facilitar a interacção com o utilizador. No entanto, a partir do momento que a lista seja acedida, os elementos consideram-se dados como notificados.

5 Conclusões e perspectivas de trabalho futuro

A metodologia de desenvolvimento de *software* foi condignamente revista pela equipa de desenvolvimento. O arranque do projecto caracterizou-se por utilizar uma metodologia mais primitiva e sem grande orientação, com procura de satisfação imediata de resultados, formando a base necessária ao desenrolar do projecto. Evoluiu para uma metodologia mais definida e com objectivos mais claros, tanto para a equipa como para o cliente (InCI). A produção de processos está pensada de forma a serem reutilizados o maior número de blocos entre os processos quanto possível. Isto é alcançado através da criação de *User Controls* que facilmente podem ser reutilizados sem grandes ajustes de maior, o que resulta num benefício claro para todo o desenvolvimento do projecto.

A ideologia do projecto passa pelo desenvolvimento de *software* utilizando uma arquitectura orientada a serviços. Este tipo de abordagem garante, conforme já foi referido anteriormente, bons resultados, com um conjunto de benefícios que assentam em redução de custos e tempo, com reutilização a níveis bastante satisfatórios, numa agilidade fulcral especialmente (e como é o caso do InCI) em ambiente onde frequentemente as regras de negócio são alteradas. Os serviços que constituem esta arquitectura são divididos em blocos que comunicam entre si, resultando num *output*. A segurança é uma questão fundamental em qualquer empresa, com a utilização desta arquitectura estão também assegurando níveis adequados.

Os produtos e tecnologias que compõem a solução são adequados ao problema e cobrem as necessidades que o projecto tem. O desenvolvimento continuado sobre a Framework 3.0 do ASP.NET, utilizando a ferramenta Visual Studio, tem resultados bastante positivos com uma optimização do tempo gasto para fazer serviços. De salvar sempre a enorme “comunicação” entre os vários produtos, como seja, Visual Studio e SQL Server 2005 ou Office Sharepoint Server 2007 e Office 2007, que se trata de uma característica que a Microsoft nos tem vindo saudavelmente a habituar.

O MOSS representa uma enorme ajuda enquanto ferramenta base de uma empresa, fornecendo um gestor de conteúdos abrangente e funcionalidades de procura, facilitando a partilha de informações dentro e fora da organização. Podendo albergar conjuntamente aplicações de internet e intranet da empresa. Este servidor de gestão de conteúdos e de colaboração fornece aos seus utilizadores e programadores a plataforma e ferramentas necessárias à correcta administração do servidor.

A necessidade de *workflows* era conhecida no projecto, como tal a Microsoft indicou que o *Windows Workflow Foundation* seria a plataforma a ser utilizada para cobrir essa necessidade.

A arquitectura lógica do projecto está de acordo com os princípios do SOA, disponibilizando de forma equilibrada, pelas várias camadas de serviços, um conjunto de infra-estruturas e serviços aplicativos adequados.

O projecto tinha como plataforma base o MOSS, pelo que foi implementado um modo de trabalho que faz a integração rápida entre os desenvolvimentos feitos no ASP.NET e o MOSS. A *Peça Web* que aqui se fala serviu convenientemente os interesses da equipa de desenvolvimento.

O *Windows Workflow Foundation* surgiu como resposta a uma necessidade dos processos do InCI, como tal foi necessário construir uma prova de conceito sobre esta plataforma, avaliando a sua viabilidade no projecto. Esta plataforma respondeu efectivamente às

necessidades do projecto. Ao aluno coube a missão de construir a prova de conceito procurando generalizar o maior número de serviços comuns. Estes serviços são largamente utilizados por todo o projecto, nomeadamente ao nível da construção de novos *workflows*, que viam a sua implementação reduzida a pouco mais do que a construção da lógica de negócio interna ao processo. A prova de conceito conseguiu estabelecer o padrão para a comunicação entre os ambientes (*workflows* e .NET), e inclui uma actividade que permite os *workflows* serem chamados assincronamente.

O projecto envolve também uma área de negócio puro, i.e. construção de processos para o negócio do InCI.

O primeiro processo implementado (Marcação de Férias do DAFRH), tinha obviamente o objectivo de obter o processo a funcionar, mas também serviu como um primeiro *input* à equipa de desenvolvimento, nomeadamente ao gestor de projecto, ajudando a despistar algumas dúvidas acerca do processo de desenvolvimento. Este processo teve por isso um papel preponderante no desenrolar do projecto.

O processo de concessão do DQ consistiu de um conjunto de funcionalidades desenvolvidas, trabalhando sobretudo com o objectivo de verificar a adequabilidade da arquitectura escolhida para o projecto. Estas funcionalidades estão a funcionar de acordo com o pretendido pelo InCI, enquanto lógica de negócio. Ao obedecer aos princípios do SOA essas funcionalidades garantiram adequabilidade para o resto do projecto, quando fosse caso de serem (re)utilizadas.

A possibilidade de trabalhar com o MOSS foi sempre um desafio muito interessante para o autor do documento. Por se tratar de uma tecnologia muito recente e com enormes potencialidades, este servidor de gestão de conteúdos foi utilizado de forma a poder facilitar algumas tarefas que eram essenciais ao projecto. A gestão documental afigurou-se como uma funcionalidade de prioridade vital ao projecto, como tal, foi desenvolvida uma *feature* de Sharepoint onde os tipos de documentos necessários à empresa foram contemplados. Estes documentos e respectivas propriedades servem convenientemente o projecto. A possibilidade de interligação entre o MOSS e os dados que figuram na base de dados do projecto (recorrendo ao *Business Data Catalog*) foi também uma característica que se revelou vantajosa no que se trata na interligação entre as propriedades dos documentos e esses dados.

Um dos principais objectivos do InCI era a desmaterialização dos processos, dado que tinha até aqui sempre um conjunto de informação que era associada aos processos. A geração de documentos confere ao projecto uma potencialidade extra, fornecendo meios para visualizar a informação em documentos próprios.

O módulo de Alertas reúne um conjunto de funcionalidades para a criação e manutenção de alertas. Os processos podem facilmente criar uma notificação que é exibida comodamente numa lista pessoal do utilizador. Os alertas fornecem um maior acesso à informação aos utilizadores, levando a que estes tenham activamente conhecimento de tarefas a executar ou outras notificações. A listagem fornece ao utilizador várias acções que são bastante úteis para a correcta manutenção dos alertas pessoais.

Ao longo do projecto surgiram algumas dificuldades, nomeadamente com o desenvolvimento dos *workflows*, onde o *designer* que integra o Visual Studio nem sempre tem um comportamento constante, com muitos períodos em que a aplicação deixa de responder ou mesmo resultando em erros, o que resultava num atrito desnecessário e incontrolável pela equipa de desenvolvimento.

As várias áreas que foram desenvolvidas conseguiram responder de forma adequada ao que se proponham. Algumas dessas áreas têm uma valia extra para a instituição de estágio, porque fornecem uma base para situações futuras em que as mesmas tecnologias ou funcionalidades sejam necessárias.

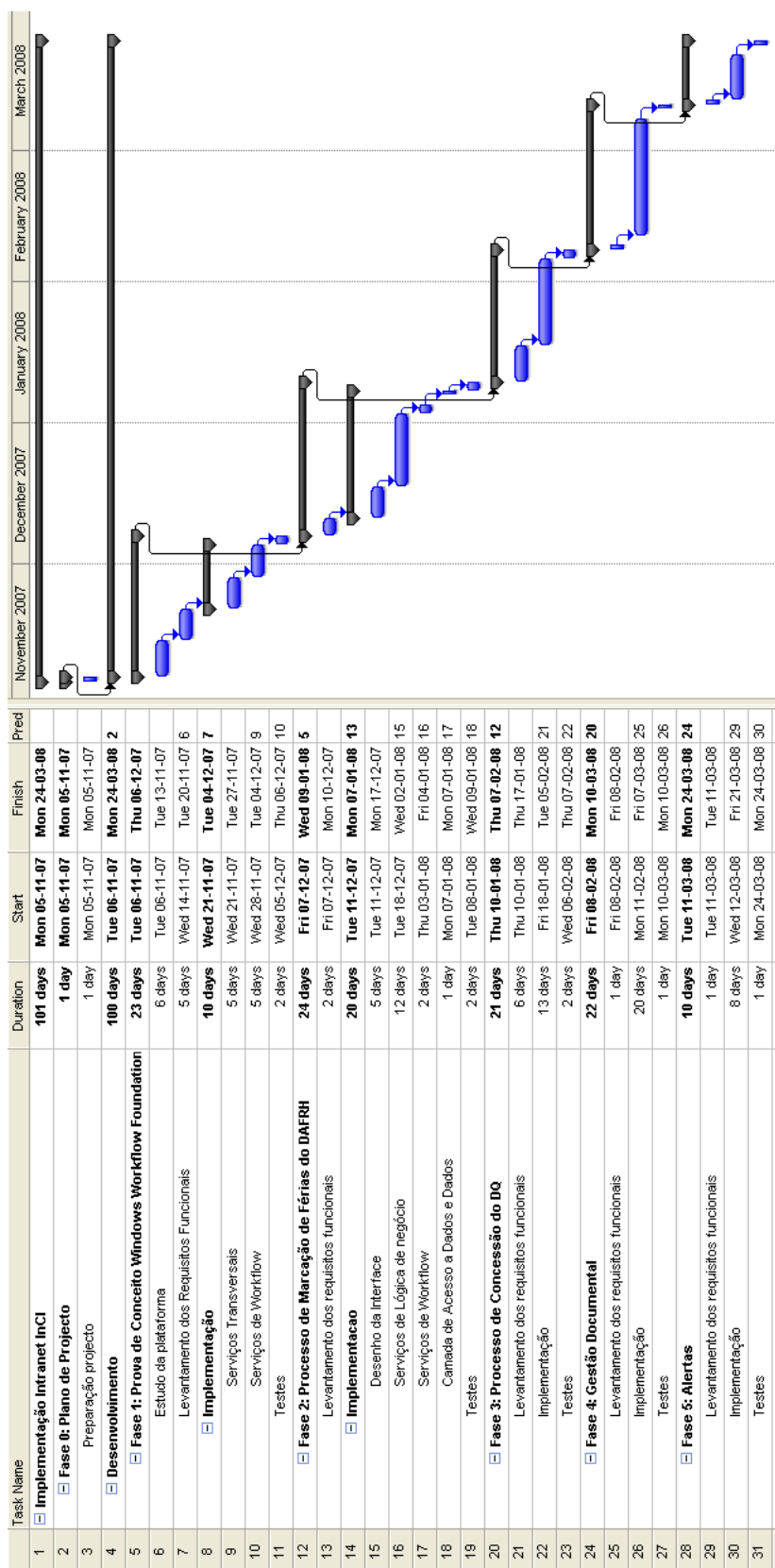
O trabalho correu na generalidade dentro do esperado, com a obtenção de funcionalidades a bom ritmo, utilizando uma abordagem que se enquadra no estudo realizado, tendo especial preponderância o facto de o trabalho ter sido realizado com tecnologias bastante recentes e adequadas ao problema.

Como trabalho futuro pode-se identificar várias áreas a ser desenvolvidas, pois o conjunto de áreas em que o aluno esteve envolvido para o desenvolvimento deste projecto enquadra-se num universo extenso de funcionalidades, fazendo parte da primeira fase do projecto total. Sendo uma das evoluções possíveis, para as funcionalidades que foram desenvolvidas pelo aluno, a criação de uma forma expedita dos *workflows* persistirem quando têm inserido nas suas actividades tempos de espera por nova informação de entrada (*DelayActivity*), que por vezes são extensos (vários dias) e os *workflows* não podem ficar em memória a aguardar.

6 Bibliografia

- [1] **Jeffrey Hasan, Mauricio Duran.** *Expert Service-Oriented Architecture in C# 2005*. 2. s.l. : Apress, 2006. 159059701X.
- [2] **Ribeiro, António.** *Arquitetura Técnica Plataforma de Serviços do InCI*. 2007.
- [3] **Bukovics, Bruce.** *Pro WF - Windows Workflow in .NET 3.0*. s.l. : Apress, 2007.
- [4] **Powell, Gavin.** *Beginnig Database Design*. s.l. : Wiley Publishing, Inc., 2006.
- [5] **Scribner, Kenn.** *Microsoft Windows Workflow Foundation Step by Step*. s.l. : Microsoft Press, 2007.
- [6] **Microsoft.** Business Data Catalog Overview. *MSDN*. <http://msdn2.microsoft.com/en-us/library/ms546541.aspx>.
- [7] **Cover, Robin.** XML and Web Services In The News . *XML.org*. <http://www.xml.org/xml/news/archives/archive.03242006.shtml>.
- [8] **Stamper, Jason.** What's up with the Web Services stack? *Tecnological News, Industry Research - Computer Business Review*.
- [9] **Wilkes, David Sprott and Lawrence.** Understanding Service-Oriented Architecture. *MSDN*. <http://msdn2.microsoft.com/en-us/library/aa480021.aspx>.
- [10] **Lenz, Gunther e Moeller, Thomas.** *.NET-A Complete Development Cycle*. s.l. : Addison-Wesley Professional, 2003.
- [11] **Holliday, John et al.** *Professional Sharepoint 2007 Development*. s.l. : Wiley Publishing, Inc., 2007.
- [12] **Kevin Hoffman, Robert Foster.** *Microsoft SharePoint 2007 Development Unleashed*. s.l. : Sams, 2007.
- [13] **Edite Estrela, Maria José Leitão, Maria Almira Soares.** *Saber Escrever uma Tese e Outros Textos*. s.l. : Dom Quixote, 2006.
- [14] **Kitta, Todd.** *Professional Windows Workflow Foundation*. s.l. : Wrox Press, 2007.
- [15] **Sun Microsystems, Inc.** *Simplified Guide to the Java™ 2 Platform, Enterprise Edition*, 1999

ANEXO A: Detalhe do planeamento do projecto



ANEXO B: Atributos de documentos

GRUPO (PERFIL BASE)	ATRIBUTOS
Entradas	Referência única do documento
	Referência da Correspondência
	Data de Entrada do Documento
	Origem
	Classificação de Entrada
	Tipos de Documento
	Assunto
	Referência Externa
	Departamento Destinatário
	Confidencial
	Entidade
	Empresa
	Pessoa
	Processo
	Ofício
	Observações
	Registado Por
	Data Registado Por
	Alterado Por
	Data Alterado Por
	Distribuido A
	Data de Distribuição
	Recepcionado Por
	Data Recepção
	Integrado Por
	Data Integração
	Data Resposta
	Integrado (S/N) -- integrado num

	processo
	Situação
	Situação Utilizador
	Situação Data
Saídas	Referência única do documento
	Referência Interna
	Data de Saída do Documento
	Origem
	Fax
	Email
	CTT
	Upload
	Classificação de Saída
	Tipos de documento
	Factura
	Reclamação
	Assunto
	Departamento Emissor
	Confidencial (S/N)
	Entidade
	Empresa
	Pessoa
	Processo
	Ofício
	Observações
	Emitido Por
	Data Emitido Por
	Enviado Por (Expediente)
	Data Enviado Por (Expediente)
	Número do Registo CTT
	Data do Registo CTT
Internos	Referência única do documento
	Número de Funcionário

	Classificação de Documento
	Tipos de documento
	Factura
	Reclamação
	Assunto
	Departamento Emissor
	Departamento Receptor
	Confidencial
	Observações
	Emitido Por
	Data Emitido Por
	Recebido Por
	Data Recebido Por