Universidade de Aveiro
Departamento de Electrónica, Telecomunicações e Informática,
2013

Fábio José
Reis Luís Marques

Arquitetura para execução segura de workflows de eGovernment dinâmicos

Fábio José
Reis Luís Marques

Arquitetura para execução segura de workflows de eGovernment dinâmicos

Tese apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Doutor em Engenharia Informática, realizada sob a orientação científica de João Gonçalo Gomes de Paiva Dias, Professor Coordenador da Escola Superior de Tecnologia e Gestão de Águeda da Universidade de Aveiro e André Ventura da Cruz Marnoto Zúquete, Professor Auxiliar do Departamento de Electrónica, Telecomunicações e Informática da Universidade de Aveiro.

Para a Ana, a Adriana e a Clara

o júri / the jury

presidente / president

Doutor Joaquim José Borges Gouveia
Professor Catedrático da Universidade de Aveiro
(por delegação do Reitor da Universidade de Aveiro)

vogais / examiners committee

Doutor José Luís Guimarães Oliveira
Professor Associado da Universidade de Aveiro

Doutor João Gonçalo Gomes de Paiva Dias
Professor Coordenador da Escola Superior de Tecnologia e Gestão de Águeda da Universidade de Aveiro

Doutora Maria Dulce Pedroso Domingos
Professora Auxiliar da Faculdade de Ciências da Universidade de Lisboa

Doutor André Ferreira Ferrão Couto e Vasconcelos
Professor Auxiliar do Instituto Superior Técnico da Universidade Técnica de Lisboa

Doutor André Ventura da Cruz Marnoto Zúquete
Professor Auxiliar da Universidade de Aveiro

agradecimentos / acknowledgements

Aos Professor Doutor Gonçalo Paiva Dias e Professor Doutor André Zúquete, pela orientação científica e sentido crítico, pela sua constante disponibilidade e amizade.

À Reitoria da Universidade de Aveiro e à Direção da Escola Superior de Tecnologia e Gestão de Águeda (ESTGA), por terem criado as condições que permitiram que usufruísse de dois anos de dispensa de serviço para me dedicar à realização deste trabalho.

À Ana pelo apoio incondicional, incentivo e companheirismo, pela sua paciência e compreensão.

Ao Hélder e ao Mário pelo companheirismo, amizade. Pelo interesse demonstrado pelo tema do trabalho, pelos seus comentários e pontos de vista.

Aos meus familiares por todo o seu apoio, disponibilidade e compreensão pelas inúmeras ausências.

A todos os colegas e amigos que, de alguma forma, mostraram interesse pelo tema do trabalho e pelo seu incentivo e apoio.

Muito obrigado!

palavras-chave

Governo eletrónico, integração de serviços, segurança, *workflows* dinâmicos, arquitetura de integração

resumo

A integração de serviços na perspetiva dos cidadãos e empresas e a necessidade de garantir algumas características da Administração Pública como a versatilidade e a competitividade colocam alguns constrangimentos na conceção das arquiteturas de integração de serviços. Para que seja possível integrar serviços de forma a que se garanta a mutabilidade da Administração Pública, é necessário criar dinamicamente *workflows*. No entanto, a criação de dinâmica de *workflows* suscita algumas preocupações ao nível da segurança, nomeadamente em relação à privacidade dos resultados produzidos durante a execução de um *workflow* e em relação à aplicação de políticas de controlo de participação no *workflow* pelos diversos executores do mesmo.

Neste trabalho apresentamos um conjunto de princípios e regras (arquitetura) que permitem a criação e execução de *workflows* dinâmicos resolvendo, através de um modelo de segurança, as questões referidas. A arquitetura utiliza a composição de serviços para dessa forma construir serviços complexos a que poderá estar inerente um *workflow* dinâmico. A arquitetura usa ainda um paradigma de troca de mensagens-padrão entre os prestadores de serviços envolvidos num *workflow* dinâmico. O modelo de segurança proposto está intimamente ligado ao conjunto de mensagens definido na arquitetura.

No âmbito do trabalho foram identificadas e analisadas várias arquiteturas e/ou plataformas de integração de serviços. A análise realizada teve como objetivo identificar as arquiteturas que permitem a criação de *workflows* dinâmicos e, destas, aquelas que utilizam mecanismos de privacidade para os resultados e de controlo de participação dos executores desses *workflows*.

A arquitetura de integração que apresentamos é versátil, escalável, permite a prestação concorrente de serviços entre prestadores de serviços e permite criar *workflows* dinâmicos. A arquitetura permite que as entidades executoras do *workflow* decidam sobre a sua participação, decidam sobre a participação de terceiros (a quem delegam serviços) e decidam a quem entregam os resultados. Os participantes são acreditados por entidades certificadores reconhecidas pelos demais participantes. As credenciais fornecidas pelas entidades certificadoras são o ponto de partida para a aplicação de políticas de segurança no âmbito da arquitetura.

Para validar a arquitetura proposta foram identificados vários casos de uso que exemplificam a necessidade de construção de *workflows* dinâmicos para atender a serviços complexos (não prestados na íntegra por uma única entidade). Estes casos de uso foram implementados num protótipo da arquitetura desenvolvido para o efeito. Essa experimentação permitiu concluir que a arquitetura está adequada para prestar esses serviços usando *workflows* dinâmicos e que na execução desses workflows os executores dispõem dos mecanismos de segurança adequados para controlar a sua participação, a participação de terceiros e a privacidade dos resultados produzidos no âmbito dos mesmos.

keywords eGovernment, service integration, security, dynamic workflows, integration architecture

abstract The integration of services from the citizens and businesses perspective and the need ensure some characteristics of the Public Administration as versatility and competitiveness puts some constraints on the design of architectures for service integration. To be able to integrate services in order to ensure the mutability of the Public Administration the creation of dynamic workflows is required. However, the creation of dynamic workflows raises some concerns in terms of security, particularly in relation to the privacy of results produced during the execution of the workflow and in relation to the application of control policies in workflow participation by many of the workflow executioners.

We present a set of principles and rules (architecture) that enable the creation and execution of dynamic workflows providing a security model that allows to solve the security issues mentioned before. The architecture combines the composition of services to construct complex services to which may be inherent a dynamic workflow. The architecture also uses a paradigm of standard messages exchange amongs service providers involved in a dynamic workflow. The proposed security model is closely linked to all the messages defined in the architecture.

Within the scope of this work several architectures and/or platform for service integration were identified and analyzed. The analysis aimed to identify the architectures that create dynamic workflows, and of these, those which use privacy mechanisms for the results and participation control by the executioners of these workflows.

The service integration architecture that we present is versatile, scalable, allows the provision of services between competing service providers and creates dynamic workflows. The architecture allows workflow participants to decide about their participation, decide on the participation of third parties (to whom they delegate services) and to whom the results are delivered. The participants are accredited by the certification authorities recognized by the other participants. The credentials provided by the certification authorities are the starting point for the application of the security policies within the architecture.

To validate the proposed architecture several use cases that exemplify the need to build dynamic workflows to address complex services (not provided in full by a single entity) were identified. These use cases were implemented in a prototype developed for this purpose. This experiment showed that the architecture is suitable to provide these services using dynamic workflows and that during the execution the security mechanisms are suited to control their participation, the involvement of third parties and the privacy of the results produced.

Conteúdo

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

A utilização das Tecnologias da Informação e da Comunicação (TIC) está a alterar a forma como o mundo se relaciona, trabalha e faz negócios. Os governos têm a necessidade de se manter a par desta revolução e utilizar as TIC em benefício da sociedade. A Administração Pública (AP) necessita de melhorar a qualidade dos serviços, de ser mais produtiva, de reduzir os custos de exploração, de corresponder às expectativas dos cidadãos e empresas, de melhorar o relacionamento com os clientes, e de se tornar um facilitador para o desenvolvimento económico [32]. Estas necessidades estão a tornar os governos mais ativos na adoção e exploração destas tecnologias (TIC) [32, 40], o que potencia o Governo Eletrónico (ou *eGovernment*).

Uma definição estável para Governo Eletrónico ainda não foi conseguida. A prova deste facto é a multiplicidade de definições que coexistem, dadas por diferentes organizações e autores. Uma pequena compilação pode ser encontrada em [69, pp.11-12]. Todavia, todas as definições partilham uma ideia base comum: a utilização das TIC para melhorar o governo.

Uma definição bastante simples de Governo Eletrónico é dada por Aniyan Varghese em [66, p.XV]: utilizar as TIC para melhorar os serviços públicos para cidadãos e empresas. Apesar desta definição ser bastante simples, o Governo Eletrónico é uma área que agrega conhecimento de várias áreas de investigação [68, pp. 11-14]: Ciências da computação; Ciências da Informação e da investigação do conhecimento; Ciências sociais e humanas; Ciências organizacionais, públicas e económicas; e, Ciências políticas e legais. A nossa contribuição enquadra-se na área de investigação das Ciências da Computação.

Segundo Ndou [46] o alvo do governo eletrónico abrange quatro grupos principais: cidadãos, empresas, governo e funcionários públicos. Daqui advêm as quatro formas de interação entre o governo (ou, na aceção latina, Administração Pública) e cada um dos grupos identificados:

1. *Government to Citizen* (G2C): carateriza-se essencialmente pelo relacionamento entre o governo e os cidadãos. Permite que os cidadãos acedam às informações e aos serviços disponibilizados pela AP.

2. *Government to Business* (G2B): refere-se ao relacionamento eletrónico entre os organismos governamentais e as empresas, reduzindo a burocracia e simplificando os processos regulatórios, permitindo, desta forma, a redução de custos de contexto e, consequentemente, facilitando a competitividade das empresas.

3. *Government to Government* (G2G): consiste na relação entre os organismos governamentais, tanto aos níveis local, regional e nacional, como ao nível internacional. A colaboração e a cooperação é essencial de forma a fornecer serviços integrados.

4. *Government to Employees* (G2E): refere-se ao relacionamento entre o governo e os seus funcionários, permitindo o acesso a informação direcionada aos funcionários da AP, tais como políticas de benefícios, oportunidades de aprendizagem, leis de direito civil, etc. A gestão de recursos humanos, orçamentação e gestão contabilística são outros dos pontos que estão diretamente relacionados com esta forma de interação.

A nossa contribuição é aplicável a todas as formas de interação enunciadas.

1.1 Motivação

Em muitos casos, o desejo de modernizar a AP levou a que diferentes ramos da AP definissem e construíssem os seus próprios sistemas de informação. Estes sistemas foram criados com objetivos específicos e com modelos de dados distintos, o que dificulta a partilha de informação entre os vários organismos da AP. Com o passar dos anos, a utilização destes sistemas isolados tornou-se insuficiente; clientes e governos queriam e necessitavam de mais [32].

A AP é composta por diversos organismos que estão organizados entre si de forma hierárquica. Cada organismo concentra-se nos serviços que lhe estão diretamente relacionados. A maioria dos serviços requeridos pelos clientes necessitam que os organismos da AP colaborem entre si. No entanto, uma vez que os organismos tendem a organizar-se de acordo com a sua área de intervenção e não de forma a cooperarem [32], os clientes têm a necessidade de aceder a diversos organismos da AP de forma a resolverem uma questão.

1.1.1 Eventos da vida: integração de serviços

A visão centrada no cliente é o paradigma que coloca o cliente no centro de toda a atividade: os serviços devem estar disponíveis sempre que o cliente necessita deles, a partir de qualquer local e de qualquer canal [31]. Uma das formas de organização dos serviços é orientá-los para *eventos da vida* (viajar, casar, ter um filho, mudar de casa, etc.), o que permite disponibilizar num único local (Portal Web, balcão único, quiosque, etc.) um conjunto de serviços da AP relacionados com o mesmo assunto [67]. Isto evita o acesso a diversos ramos do governo para reunir a informação relevante para solicitar um serviço, potencia uma diminuição dos custos para os cidadãos e a diminuição do tempo gasto para obter o serviço.

A integração de serviços do ponto de vista dos clientes, permitida pelos sistemas de informação da AP, tem muitas vantagens para o próprio governo e para os seus clientes. Os benefícios para os clientes são óbvios, uma vez que toda a informação necessária pode estar disponível quando e onde é necessária, a partir de um vasto leque de canais: balcões de atendimento presencial; computadores pessoais; quiosques; telemóveis; *Personal Digital Assistant* (PDA); *call centers*; etc. Há também vantagens do lado do governo: as comunicações entre ramos do governo diminuem o tempo de prestação do serviço, aumentam a produtividade e, consequentemente, diminuem os custos. A integração de serviços pode, igualmente, ser um incentivo para repensar os processos, tornando a prestação de serviços mais eficiente.

A integração de serviços não está limitada aos serviços oferecidos pelo governo; também pode ser estendida às entidades privadas. Por exemplo, imagine-se que o Manuel e a Maria vão mudar de residência. Para que mantenham os seus registos atualizados, acedem ao portal do governo e alteram a sua morada em todos os sistemas governamentais num único passo. Através do mesmo portal, eles podem utilizar a mesma plataforma de integração de serviços

para aceder a um conjunto de prestadores de serviço (água, eletricidade, etc.) e alterar os seus contratos da residência antiga para a nova.

Para atingir este objetivo têm sido propostas plataformas de integração de serviços (e.g. Projeto eGov [70], Plataforma de Integração [2]). Algumas destas plataformas de integração foram financiadas por iniciativas que tinham como missão o desenvolvimento de soluções para resolver a questão da interoperabilidade [23, 57]. Os programas *Interoperability Solutions for Public Administrations* (ISA)[1] (precedido pelos programas designados por *Interoperable Delivery of European eGovernment Services to public Administrations, Businesses and Citizens* (IDABC) e por *Interchange of Data between Administration* (IDA)) da União Europeia e o programa *Media@komm-Transfer*[2] (que sucede o programa *Media@komm*) do Ministério Federal da Economia da Alemanha são exemplos deste tipo de iniciativas.

1.1.2 Competição

Vários organismos da AP disponibilizam serviços similares (e.g. ensino, saúde, notariado, etc.). Adicionalmente, existem entidades privadas que estão aptas a fornecer serviços públicos (e.g. escolas privadas, clínicas e hospitais privados, etc.), aumentando a gama de opções para a prestação de um serviço. Isto significa que um cliente da AP tem, muitas vezes, a possibilidade de escolher a entidade a quem vai solicitar a prestação do serviço de que necessita no âmbito de um evento da vida.

1.1.3 Versatilidade

A AP está em constante mutação: organismos que a compõem deixam de existir e são criados outros; novas leis são criadas e outras deixam de existir. Tendo em consideração que nem todos os serviços são disponibilizados digitalmente em simultâneo, que os serviços podem ser adicionados e removidos, que os organismos têm tempos de vida distintos, é necessário garantir a versatilidade na prestação de serviços complexos orientados para os eventos da vida.

Para que seja possível integrar serviços, serviços estes que podem ser fornecidos por diferentes prestadores de serviços, e de forma a que se garanta a mutabilidade da AP, necessitamos criar dinamicamente *workflows* [15] (*workflow* cuja sequência de tarefas é definida durante a execução). A capacidade de criar dinamicamente *workflows* permite que o *workflow* gerado tenha em consideração as preferências do cliente e a necessidade das entidades, públicas ou privadas, envolvidas no processo. Por exemplo: requerer licença de publicidade a uma câmara municipal; a determinada altura da avaliação do processo poderá ser necessário solicitar um parecer a um organismo externo, o organismo a consultar depende das condicionantes do processo e que apenas são conhecidas na altura da solicitação do parecer.

1.2 O Problema

A criação dinâmica de *workflows*, executados por um conjunto de organizações que colaboram com os seus serviços para a prestação de um serviço complexo suscita algumas preocupações ao nível da privacidade e da segurança. Estas questões estão relacionadas com o facto de

[1] `http://ec.europa.eu/isa/`

[2] `http://www.mediakomm-transfer.de/`

não ser possível identificar qual ou quais os intervenientes na prestação do serviço subjacente ao *workflow* completo no momento em que esse serviço é solicitado:

1. Como se garante que um organismo está autorizado a fornecer um determinado serviço que possa ser usado no âmbito de um serviço complexo?

2. Como se permite que um organismo decida se participa ou não na execução de um *workflow* dinâmico inerente a um serviço complexo para o qual um dos seus serviços foi solicitado?

3. Como se garante que um resultado produzido por um prestador de serviço, usado no âmbito de um workflow dinâmico, é entregue aos destinatários e apenas aos destinatários corretos quando estes não são conhecidos na altura em que o resultado é produzido?

1.3 Objetivos

O principal objetivo do nosso trabalho é justificar, definir, validar e discutir um conjunto de princípios e regras (arquitetura) para resolver as questões de segurança associadas à execução de *workflows* dinâmicos no contexto do governo eletrónico. A realização deste objetivo implica a concretização de objetivos complementares: o levantamento das questões de segurança associadas à execução de *workflows* dinâmicos; definição de um modelo de segurança que aborde as questões de segurança identificadas; o desenvolvimento de um protótipo de plataforma que respeite a arquitetura; e a identificação e a implementação nesse protótipo de casos de uso que permitam a validação da arquitetura proposta.

1.4 Contribuição

A principal contribuição deste trabalho é um conjunto de princípios e regras (arquitetura) que permitem resolver as questões de segurança levantadas pela execução de *workflows* dinâmicos em plataformas de governo eletrónico. A arquitetura permite que os participantes, que designamos por Entidade (ver Figura 1.1), no ato de prestação de um serviço no âmbito de um *workflow* complexo envolvendo várias Entidades, decidam sobre a sua participação, a quem vão delegar parte do serviço e a quem entregam os resultados.

Uma entidade atua como um representante de um organismo. A inserção, atualização e remoção de serviços e entidades resume-se à publicação e remoção da descrição de serviços em repositórios de serviços. As entidades são acreditadas para o fornecimento de um serviço por entidades certificadoras que são reconhecidas pelas demais entidades. As credenciais fornecidas pelas entidades certificadoras às entidades são o ponto de partida para a operação da arquitetura relativamente à autenticação de entidades, autenticação da informação que produzem, controlo de acesso de outras entidades aos serviços prestados por uma entidade, controlo da participação de uma entidade num *workflow* em curso e entrega de resultados apenas à entidade correta.

Para validar a arquitetura proposta foram identificados vários casos de uso que exemplificam a necessidade de uso de *workflows* dinâmicos e esses casos de uso foram concretizados num protótipo desenvolvido para o efeito. O protótipo foi implementado utilizando a plataforma de agentes de software *Java Agent DEvelopment Framework* (JADE). O mesmo foi desenvolvido

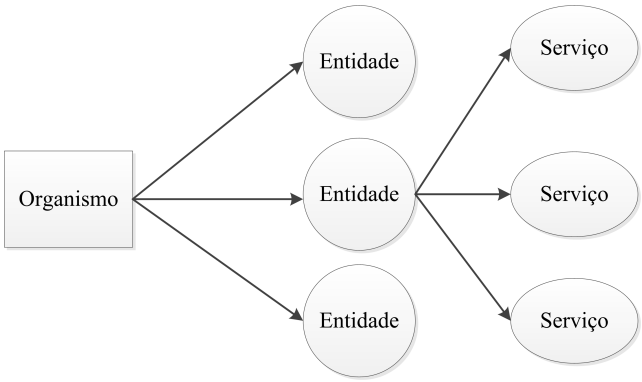

Figura 1.1: Representação da relação entre Organismo, Entidades, e Serviços.

com recurso à linguagem Java, implementa as estruturas de dados e simula os serviços prestados pelos organismos da AP. Este protótipo recorre ao *Workflows and Agent Development Environment* (WADE), um plug-in do JADE que permite a definição de *workflows* tornando possível a definição do comportamento interno das Entidades.

Os casos de uso escolhidos para a validação são os seguintes: consulta médica; candidatura ao ensino superior; processo de permuta de imóvel; e, viagem ao estrangeiro. Estes casos de uso foram escolhidos devido às suas características, permitindo exemplificar a aplicação da arquitetura em diferentes contextos. Todos os casos de uso foram adaptados para permitir a integração de serviços no âmbito da execução de um *workflow* dinâmico, pelo que não refletem a forma como os serviços são prestados atualmente.

1.5 Publicações

Durante o decurso do trabalho de doutoramento subjacente a esta dissertação foram sendo publicados alguns resultados intermédios. Nesta secção vamos apresentá-los cronologicamente:

- Em 2009, publicámos um trabalho cujo principal contributo relaciona-se com a identificação e análise das principais questões de segurança que têm a sua origem na composição dinâmica de serviços prestados por agentes de organismos. Nesta comunicação foram identificadas possíveis abordagens às diferentes questões de segurança levantadas.

  F. Marques, G. Dias, e A. Zúquete. Security concerns in e-Governmentk agent-based interoperability. In *Proceedings of ongoing research, general development issues and projects of EGOV 09, 8th International Conference*, páginas 197-204, 2009.

- Em 2011, publicámos um artigo onde apresentámos a arquitetura proposta nesta tese e a sua concretização usando agentes e Web Services. Neste trabalho identificámos e descrevemos as principais componentes da arquitetura.

  F. Marques, G. Dias, e A. Zúquete. A general interoperability architecture for e-Government based on agents and Web Services. In *6ª Conferência Ibérica de Sistemas e Tecnologias de Informação*, páginas 338-343, 2011.

- Em 2012, publicámos um artigo que tem o seu enfoque no modelo de segurança utilizado na arquitetura. Apresenta todas as estruturas de suporte que permitem o controlo de

acesso a serviços e resultados e a acreditação de pares {entidade, serviço}. Uma das principais novidades no modelo é o facto de o resultado apenas sair da entidade que o produz quando explicitamente solicitado.

F. Marques, G. Dias, e A. Zúquete. Modelo de segurança para a composição dinâmica de workflows em arquiteturas de e-government. In *Revista Ibérica de Sistemas e Tecnologias de Informação*, 9: 81-90, 2012.

- Em 2013, publicámos um artigo que apresenta o protótipo que implementa a arquitetura apresentada nesta tese. Neste trabalho discutimos a escolha das tecnologias utilizadas na implementação de um protótipo que implementa a arquitetura, apresentamos o protótipo e um caso de teste (viagem a um país estrangeiro).

  F. Marques, G. Dias, e A. Zúquete. Agent-based Interoperability for e-Government. In *10^{th} International Symposium on Distributed Computing and Artificial Intelligence (DCAI'13)*, in print, 2013.

- Estamos a concluir um novo trabalho a submeter para a revista *Electronic Commerce, Research and Applications*. Nesse artigo identificamos a necessidade de utilização da execução de *workflows* dinâmicos, descrevemos a arquitetura e o protótipo, analisamos um conjunto de arquiteturas de integração de serviços e fazemos a sua análise relativamente à execução de *workflows* dinâmicos e à segurança. Por fim, discutimos as soluções apresentadas, comparando-as com a nossa abordagem.

  F. Marques, G. Dias, e A. Zúquete. Privacy Preserving Architecture for e-Government Dynamic Workflows.

1.6 Organização

Esta dissertação está subdividida em cinco capítulos. No primeiro capítulo identificámos o problema, definimos os objetivos, apresentámos a contribuição da investigação, indicámos as publicações realizadas e previstas no âmbito do trabalho e descrevemos a organização do documento. Seguem-se os restantes quatro capítulos:

- No capítulo 2, começamos por definir a terminologia utilizada em todo o documento. Em seguida, identificamos os quadros de referência para a interoperabilidade mais relevantes para este trabalho. Procedemos com a identificação dos paradigmas relevantes para a nossa contribuição e fazemos o levantamento, descrição e análise de um conjunto de arquiteturas e plataformas de integração de serviços para governo eletrónico. Segue-se o levantamento de um conjunto de modelos de controlo de acesso que são relevantes no âmbito do nosso contributo. Este capítulo conclui-se com uma resenha das principais conclusões.

- No capítulo 3, começamos por identificar os objetivos que pretendemos que a nossa solução atinja. Na secção seguinte, identificamos os problemas de segurança que advêm da concretização dos objetivos definidos. Em seguida, expomos a nossa solução, apresentando as ideias gerais da arquitetura e exemplificamos, com base num caso de uso. O modelo de segurança é descrito posteriormente. Em seguida, descrevemos a arquitetura e as suas componentes (nomeadamente, os serviços e estruturas de suporte, as mensagens e as entidades). Por fim, discutimos a arquitetura analisando-a de acordo com os objetivos identificados e concluímos o capítulo com um sumário dos resultados obtidos.

- No capítulo 4, validamos a arquitetura proposta no capítulo anterior. Para esse efeito usaremos um conjunto de casos de uso representativos de serviços complexos que podem ser prestados através de um *workflow* dinâmico executado por diversas organizações. Para concretizar esses casos de uso construímos um protótipo, que também é descrito neste capítulo, o qual nos permitiu validar a adequação da arquitetura proposta para lidar com esses casos de uso tendo em linha de conta as políticas de segurança adequadas.
- No capítulo 5, concluímos o documento com uma resenha do trabalho efetuado, validando os objetivos propostos para o mesmo. Fazemos igualmente algumas considerações gerais e identificamos o trabalho futuro.

Capítulo 2

Enquadramento

No capítulo anterior fizemos uma introdução ao tema, contextualizando-o e balizando-o. Neste capítulo, na secção 2.1, começaremos por definir os termos utilizados nesta tese. Em seguida, na secção 2.2, identificamos um conjunto de quadros de referência para a interoperabilidade, a sua importância para a realidade atual dos sistemas de informação na Administração Pública, as diferentes abordagens que utilizam relativamente à integração e identificamos aqueles que são particularmente relevantes para a nossa contribuição. Na secção 2.3, identificamos paradigmas relevantes para a integração de serviços. Na secção 2.4, apresentamos, descrevemos e analisamos um conjunto de arquiteturas e/ou plataformas de integração de serviços. A análise tem como finalidade identificar quais as arquiteturas e/ou plataformas de integração que permitem a utilização de *workflows* construídos dinamicamente e, nesse caso, a sua abordagem relativamente à privacidade da informação e ao controlo da participação no *workflow*. Serão igualmente identificadas as tecnologias e abordagens que as arquiteturas e/ou plataformas de integração utilizam relativamente à segurança. Na secção 2.5, identificamos os principais modelos de controlo de acesso. Por fim, na secção 2.6, concluímos o capítulo.

2.1 Terminologia

Esta secção tem como objetivo uniformizar os termos utilizados na tese. Embora alguns deles sejam utilizados indiscriminadamente na literatura. Neste documento sempre que utilizarmos algum destes termos é com o sentido que definimos nesta secção.

- Arquitetura: um conjunto de princípios e regras
- Plataforma: concretização que respeita um determinado conjunto de princípios e regras. Ou seja, a concretização de uma arquitetura.
- *Workflow*: sequência de passos ou tarefas que permite a prestação de um serviço (simples ou complexo).
- *Workflow* pré-estabelecido: *Workflow* cuja sequência de passos está definida desde o momento em que o serviço é disponibilizado.
- *Workflow* pré-construído: *Workflow* cuja sequência de passos é definida no momento em que o serviço é invocado.

- *Workflow* construído dinamicamente: *Workflow* cuja sequência de passos é definida durante a execução do *workflow*.
- Quadro de referência (*framework*) para a interoperabilidade: documento que define regras, normas e linhas orientadoras para atingir a interoperabilidade.

2.2 Quadros de referência para a interoperabilidade

No contexto do governo eletrónico, os sistemas de informação isolados e os seus modelos de dados específicos tornam a integração de serviços bastante difícil de atingir. De forma a conseguir integrar diferentes organismos da AP, diversos quadros de referência para a interoperabilidade foram criados para regular a forma como os sistemas devem interoperar. Estes quadros de referência definem regras, normas e linhas orientadoras. Um dos exemplos, a nível internacional, é o *European Interoperability Framework* (EIF) [18], que atua como um suplemento para os quadros de referência nacionais para a interoperabilidade e como um guia para a interoperabilidade ao nível dos sistemas da União Europeia.

Ao nível nacional, cada país define o seu quadro de referência, existindo, por isso, diversos quadros de referência, com maturidades distintas, para a interoperabilidade: o *Federal Enterprise Architecture Framework* (FEAF) [16], que define as regras de utilização obrigatória para a interoperabilidade nos Estados Unidos da América; o *electronic Government Interoperability Framework* (eGIF) [12] no Reino Unido; o *Standards and Architectures for e-Government Applications* (SAGA) [24] na Alemanha; e o Guia Integração Electrónica (GIE) em Portugal [2], são alguns dos exemplos. Uma compilação bastante completa de quadros de referência para a interoperabilidade pode ser encontrada em [60, Anexo B]. Estes quadros de referência utilizam diferentes abordagens relativamente à integração e ao seu âmbito.

Segundo a análise de Klischewski [37], os quadros de referência para a interoperabilidade na União Europeia e nos Estados Unidos têm duas abordagens diferentes relativamente à integração. O quadro de referência da União Europeia favorece a abordagem da integração de processos, enquanto o quadro de referência dos Estados Unidos da América favorece a abordagem da integração da informação. A abordagem da integração de processos consiste na interação entre os organismos recorrendo aos processos de cada organismo de forma a disponibilizar novos processos. Por seu turno a abordagem da integração da informação consiste no acesso facilitado à informação por parte dos organismos públicos.

De notar que dois dos pontos que os quadros de referência para a interoperabilidade têm em comum é o enfoque que colocam na segurança e a recomendação de utilização de normas abertas na implementação das arquiteturas de integração ou de sistemas que visam a interoperabilidade.

Do ponto de vista do nosso contributo, os quadros de referência para a interoperabilidade que devem ser respeitados pela nossa solução, devido ao contexto em que nos encontramos, são o EIF na União Europeia e o GIE em Portugal.

2.3 Paradigmas

Nesta secção apresentamos os paradigmas utilizados pelas diferentes arquiteturas de integração que iremos apresentar e analisar na secção seguinte. É nosso objetivo identificar as

principais características destes paradigmas. De notar que os paradigmas que apresentamos não são incompatíveis, podendo coexistir na mesma arquitetura.

2.3.1 *Service Oriented Architecture*

O paradigma da Arquitetura Orientada a Serviços é, presentemente, um dos paradigmas arquiteturais mais utilizados. A maioria dos sistemas que permitem a integração de serviços utilizam este paradigma para tornar possível a comunicação entre pares. Em [75] *Service Oriented Architecture* (SOA) é definido como um conjunto independente de serviços que são implementados como componentes que podem ser invocados, sendo as descrições das suas interfaces publicáveis, pesquisáveis e descobríveis. Este paradigma permite a criação de sistemas de aplicações interoperáveis e modulares. Em [14] é identificado um conjunto de características responsáveis pelo sucesso do SOA: escalabilidade; abstração; e utilização de normas abertas.

2.3.2 *Message Oriented Middleware*

Curry, em [19], define *Message Oriented Middleware* (MOM) como qualquer infraestrutura de *middleware* que fornece capacidades de mensagens. Esta é uma definição bastante abrangente. Uma definição mais detalhada é dada pelo Grupo Gartner [27] que define MOM como uma infraestrutura que permite a interoperabilidade através da comunicação entre aplicações. A comunicação varia entre troca de mensagens de forma totalmente assíncrona e síncrona. As principais vantagens que se podem obter de arquiteturas que utilizem este paradigma são: *loose-coupling*; fiabilidade; escalabilidade; e disponibilidade.

2.4 Arquiteturas de integração

As vantagens de utilizar as plataformas de integração como forma de prestar serviços de governo eletrónico são bastante superiores às suas desvantagens. As plataformas de integração podem ser utilizadas como catalisadores para redefinir processos, permitindo criar serviços mais eficientes. A troca de informação entre organismos do governo torna-se mais rápida, o que reduz o tempo de prestação de serviço, aumenta a produtividade, permitindo reduzir custos [32]. Os clientes também beneficiam desta melhoria na prestação dos serviços: toda a informação pode estar disponível quando e onde for necessária, a partir de um vasto conjunto de canais: escritórios da Administração Pública, computadores pessoais, quiosques, telefones móveis, PDA, *call centers*, etc.

Várias arquiteturas e/ou plataformas de integração têm sido propostas ao longo dos anos. As arquiteturas e/ou plataformas que apresentaremos em seguida, foram publicadas em atas de conferência ou revistas científicas entre 2001 e 2008, nomeadamente: IEEE Computer Society; DEXA - Electronic Government; IFIP - Knowledge Management in Electronic Government; Electronic Commerce Research and Applications; Government Information Quaterly. Adicionalmente, apresenta-se a descrição da Plataforma de Integração (PI) da Administração Pública Portuguesa cuja descrição apresentada tem como base o GIE e os sítios web oficiais da Agência para a Modernização Administrativa (AMA) e da Agência para a Sociedade do Conhecimento (UMIC).

Nas secções que se seguem descrevem-se as arquiteturas e/ou plataformas de integração encontradas, a saber: eGov Project [70]; Online Services Computer Interface (OSCI) [61]; arquitetura Dias e Rafael (D&R) [21]; Secure Electronic Contract (SeCO) Container [29]; eMayor

Project [35]; Web Digital Government (WebDG) [44]; Plataforma de integração (PI) [3]; Access-eGov [39]; AIPA architecture [6]. Estas arquiteturas e/ou plataformas serão analisadas relativamente à utilização de *workflows* construídos dinamicamente, se o permitem ou não. E, no caso de o permitirem, se preveem algum mecanismo para assegurar a privacidade dos resultados e o controlo da participação no *workflow*. Adicionalmente, analisam-se as soluções de segurança que cada arquitetura e/ou plataforma apresenta. Todas as arquiteturas e/ou plataformas que apresentamos utilizam pelo menos um dos paradigmas descrito na secção 2.3: SOA ou MOM.

2.4.1 eGov Project

O Projeto eGov (*An Integrated Platform for Realizing Online One-Stop Government*) [62, 70] foi um projeto financiado pelo 5º Programa Quadro de I&D da União Europeia (UE). A arquitetura eGov, proposta no âmbito deste projeto, segue o paradigma SOA. Teve como objetivos técnicos a especificação e desenvolvimento de: um portal para a AP como ponto único de acesso em linha; um repositório de serviços e um ambiente para a criação de serviços; e, uma linguagem *markup* - *Governmental Markup Language* (GovML) - para a descrição de serviços governamentais.

A Figura 2.1 representa a estrutura geral da arquitetura eGov. Como é possível constatar, existe um portal que representa a entrada global de acesso aos diferentes repositórios de serviços. Os clientes podem utilizar diferentes meios de acesso ao portal: computador pessoal; PDA, etc. Os serviços são disponibilizados segundo a abordagem eventos da vida e situações de negócio.

A linguagem GovML foi especificamente definida para o projeto. Permite a descrição de serviços públicos e de eventos da vida [36]. A arquitetura disponibiliza dois tipos de serviços: informativos e transacionais. Se o serviço, solicitado pelo cliente, for informativo, então a informação relevante é enviada diretamente para o portal. No caso de ser solicitado um serviço transacional, então o serviço é executado pelo *Service Runtime Environment*, que o recebe, o processa e o decompõe e redireciona os serviços resultantes da decomposição. O resultado desta prestação de serviço é enviado para o cliente através do portal.

Esta arquitetura não permite que o *workflow* seja construído dinamicamente. Os serviços do tipo transacional, quando solicitados, e embora sejam decompostos pelo *Service Runtime Environment* já têm o seu *workflow* previamente estabelecido e os organismos participantes definidos.

Relativamente à segurança, a maioria das tecnologias estudadas e utilizadas na plataforma interagem com o GovML. Para a autenticação, integridade e não-repúdio é utilizado o XML *Signature* [72]. O XML *Encryption* [76] garante a confidencialidade da informação nas mensagens trocadas e o *Secure Socket Layer* (SSL)/ *Transport Layer Security* (TLS) garantem a confidencialidade durante a comunicação.

2.4.2 Online Services Computer Interface

A plataforma OSCI [52, 61] foi desenvolvida no âmbito do projeto Media@komm[1]. Este projeto teve como objetivo principal estimular o desenvolvimento de aplicações multimédia em aldeias, cidades e comunidades locais. É uma plataforma baseada no paradigma MOM.

[1]http://mediakomm.difu.de/(versão original) e http://mediakomm.difu.de/en/index.php (versão inglesa) ambos atualmente desativados

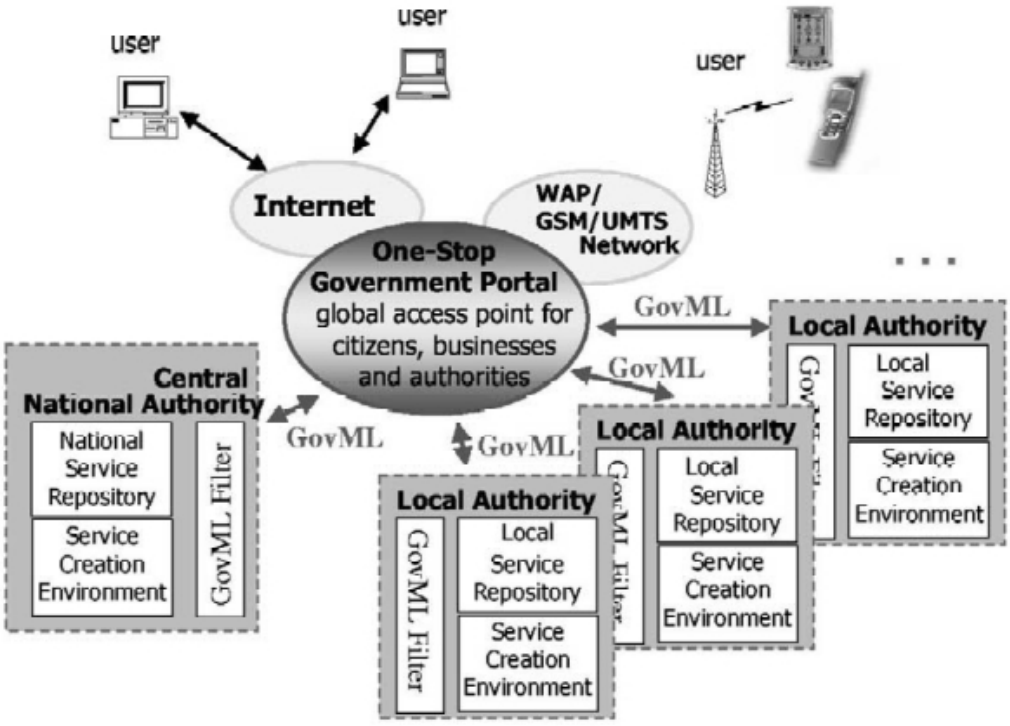

Figura 2.1: Representação da plataforma eGov (Fonte: Wimmer, 2002)

O OSCI é, por desenho, uma norma de mensagens que permite transações legalmente vinculativas. É, no momento, a norma para o transporte seguro de dados na administração pública na Alemanha. De facto, a versão 4.0 do SAGA [24] indica a utilização do protocolo OSCI-Transport versão 1.2 [51] como obrigatória a nível nacional na Alemanha. Esta norma de mensagens foi pensada para dois tipos de cenários: um cenário do tipo pedido-resposta para processos que podem ser satisfeitos de forma automática - comunicação síncrona; e para processos que requerem intervenção humana - comunicação assíncrona.

A arquitetura OSCI é essencialmente um protocolo de duas camadas: a camada de segurança e a camada aplicacional. A camada de segurança materializa-se no *OSCI-Transport* que se encontra na versão 2 [53]. Todas as funções criptográficas estão definidas nesta camada. A Figura 2.2 representa as funcionalidades e os serviços suplementares da arquitetura.

A camada aplicacional define a forma como é representado o conteúdo das mensagens a serem transmitidas, por outras palavras define uma norma (recorrendo a Esquemas *Extensible Markup Language* (XML)) para representar o conteúdo a transmitir. Isto permite a criação de diferentes aplicações que utilizam mensagens normalizadas recorrendo ao OSCI-Transport para as entregar. Uma das normas mais conhecida é o OSCI-XMeld [7] que suporta os processos relacionados com o registo de cidadãos.

A camada de segurança do OSCI foi criada para providenciar a transferência segura (de acordo com o *German Digital Signature Act* [55, SigG artigo 3]) de conteúdo entre dois parceiros de comunicação, *Source e Target Application* (ver Figura 2.2), utilizando assinaturas digitais. A camada de segurança visa diversos objetivos [51]: Interoperabilidade; independência da aplicação; escalabilidade; independência da plataforma. Para os atingir recorre a

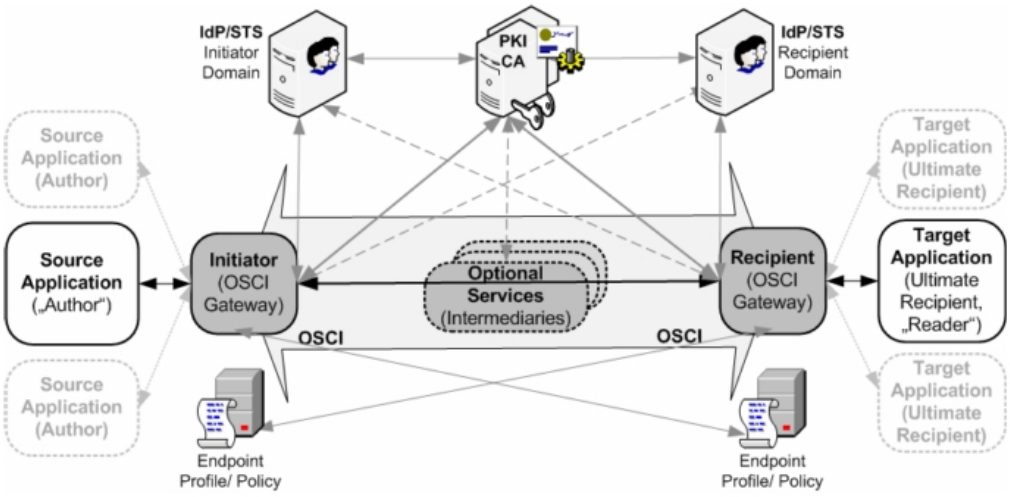

Figura 2.2: Representação da camada de segurança da arquitetura OSCI (Fonte: OSCI Steering Office, 2009)

diversas tecnologias padrão, a maior parte delas baseando-se em normas XML [71]: XML *Encryption*, XML *digital signature*, *Simple Object Access Protocol* (SOAP) [73], etc. Isto permite a realização de objetivos genéricos de segurança, como iremos analisar, e essenciais para a segurança de sistemas com as características do OSCI: autorização; confidencialidade; integridade e autenticidade; não-repúdio e identificação do caminho percorrido.

As funcionalidades atribuídas aos *OSCI Gateways* dependem da sua função (*Initiator* ou *Recipient*). Na função *Initiator*, o *OSCI Gateway* inicia a comunicação compondo os dados necessários, destacando-se: o acesso às políticas do *endpoint* do destino da mensagem; a obtenção do *token* de segurança que solicita ao *STS*; e a assinatura e cifra ao nível do transporte. Do lado do *Recipient*, o *OSCI Gateway* recebe (comunicação síncrona) ou extrai (comunicação assíncrona), da caixa de mensagens no intermediário (*Intermediary*), a mensagem enviada pelo *Initiator*, decifra a mensagem, verifica os dados da comunicação, a identificação e a autorização do *Initiator* e entrega o conteúdo e a informação da validação à *Target Application*.

Como vimos atrás, a comunicação é suportada por um intermediário, que pode realizar outros serviços para além de entregar mensagens. O intermediário atua como um ponto central de troca de mensagens, gerindo caixas de correio e distribuindo as mensagens. A principal razão da sua existência é a comunicação assíncrona.

Em [51, 53], pode ser encontrada uma descrição mais detalhada do modelo de comunicação do OSCI. As mensagens enviadas podem ter vários autores mas apenas um remetente. Estas mensagens podem ter como destino apenas um recetor mas podem ter vários leitores. Cada mensagem é composta por duas secções (ver Figura 2.3): *content data*; e *communication data*. Cada uma com um propósito específico. Na secção *content data* podemos encontrar o conteúdo resultante da contribuição dos vários autores. A secção *communication data* é utilizada para encapsular a informação necessária para processar a mensagem. Isto inclui a informação digital sobre os autores e remetentes: as assinaturas digitais; e os seus certificados de cifra. Inclui igualmente os certificados de cifra dos leitores e do destinatário, *time-stamps* e o estado do pedido.

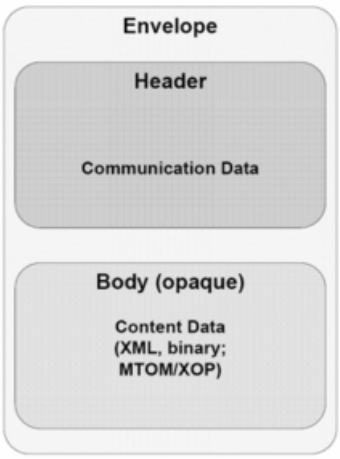

Figura 2.3: Representação da estrutura da mensagem do OSCI (Fonte: OSCI Steering Office, 2009)

A plataforma OSCI não permite *workflows* construídos dinamicamente, não foi possível obter informação sobre como os serviços são decompostos ou se efetivamente existe decomposição de serviços. Relativamente à segurança, os autores definem e abordam diversos requisitos. A autenticação é assegurada uma vez que a cada caixa de correio está associada a um destinatário através do certificado. Para aceder às suas mensagens, o destinatário, tem de se autenticar, isto é feito durante a inicialização do diálogo. Os dados são também autenticados. No caso do *content data* isto é feito pelos certificados dos autores. No caso do *communication data* isto é conseguido através do uso do certificado do remetente. Adicionalmente, a partir da versão 2 [53], o OSCI utiliza a norma *Security Assertion Markup Language* (SAML) [48, 49] para expressar as propriedades de autenticação.

A Autorização não foi um objetivo da arquitetura na versão 1.2 do OSCI-Transport. No entanto, na versão 2, a utilização da norma SAML e de *tokens* SAML permite cumprir este requisito. O *token* SAML é enviado juntamente com as mensagens, o que permite que o *OSCI Gateway Recipient* e a *Target Application* possam ter acesso às propriedades de autorização fornecidas pelo *OSCI Gateway Iniciator*.

A Confidencialidade é abordada através da cifra de ambas as secções da mensagem. O intermediário apenas tem acesso à informação contida na secção *communication data* de onde retira a informação necessária para entregar as mensagens. O conteúdo da mensagem está protegido, no caso de informação sensível, do destinatário ou de outros leitores. Desta forma apenas o leitor a que se destina o conteúdo está autorizado a lê-lo.

O conteúdo da mensagem é assinado digitalmente, o que permite garantir a integridade. Na secção *communication data* pelo remetente e na secção *content data* pelos autores. Isto permite a identificação dos dados que são alterados durante a transmissão. A alteração dos dados da secção *content data* pode ser detetada pelos leitores enquanto que a alteração dos dados da secção *communication data* podem ser detetados pelo destinatário ou pelo intermediário.

A utilização dos registos do sistema, do *timestamping* na mensagem e das assinaturas digitais permitem seguir as ações dos utilizadores, incluindo as que são realizadas por utilizadores não autorizados, assegurando a rastreabilidade. O não-repúdio é garantido pelos registos e

pelo conteúdo assinado digitalmente o que previne a negação das intervenções por parte dos participantes.

2.4.3 Contentor Secure Electronic Contracts

A plataforma SeCO Container [29, 30, 59] faz parte do projeto Secure Electronic Contracts. A plataforma utiliza o paradigma MOM. Nas fontes utilizadas não existe informação sobre a sua aplicação concreta, no entanto pode ser utilizada para permitir a interoperação entre todos os organismos da Administração Pública. Este projeto foi desenvolvido no *Media Communications Management* (mcm*institute*) da Universidade de St. Gallen e no Departamento para a Ciência de Computadores e no Departamento de Direito da Universidade de Zurique, em colaboração com a Câmara de Comércio de Zurique e o *Electronic Mall Bodensee Management AG*. A plataforma tem as suas raízes no Comércio Eletrónico, foi inspirada no *Business Media Reference Model* [58] e destina-se à negociação e concretização segura de contratos eletrónicos. É composta por três camadas (ver Figura 2.4):

(a) Lógica - desenha, gere e realiza a lógica do *workflow*;

(b) Informação - providencia espaço para armazenamento e é nesta camada que se encontra a informação sobre o contrato;

(c) Comunicação - disponibiliza as funções de segurança (que se suportam numa *Public Key Infrastructure* (PKI)).

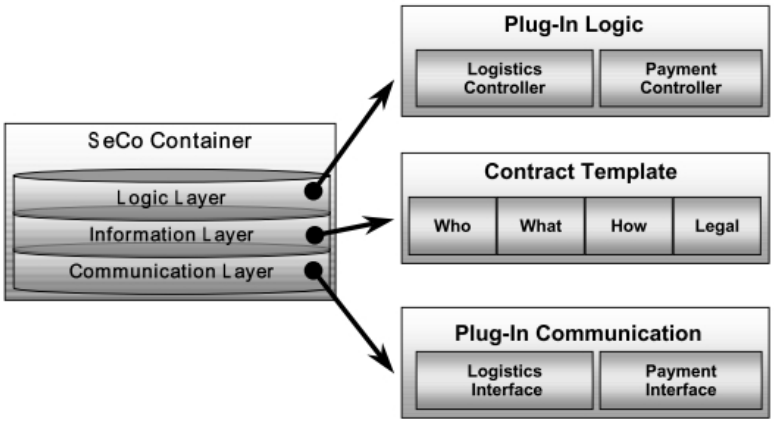

Figura 2.4: Arquitetura SeCO (Fonte: Bernd Schopp, 2000)

É na camada lógica que o desenho, a gestão e a execução do *workflow* da transação de negócio é implementado. Por outras palavras, descreve o processo para o cumprimento do contrato. Esta camada requer informação contida na camada de Informação que é acedida de forma segura.

Todos os dados que descrevem os termos e a informação do contrato são armazenados na camada de Informação. Estes dados estão divididos em duas partes: uma parte estruturada e uma parte não estruturada. A informação contida na parte estruturada está organizada em quatro blocos de informação, que permitem a caraterização completa do contrato: o bloco

Who, que descreve os intervenientes no contrato; o bloco *What*, que contém informação sobre o produto ou serviço do contrato; o bloco *How*, que contém informação sobre as condições acordadas pelas partes; e o bloco *Legal*, que contém as circunstâncias legais às quais o contrato está sujeito. Todos os documentos relevantes que não têm um formato definido são inseridos na parte não estruturada da camada de Informação.

Os protocolos que são necessários para estabelecer a comunicação entre os parceiros de contrato e os serviços de mercado podem ser encontrados na camada de comunicação. É igualmente nesta camada que as funções de segurança para a troca de mensagens se encontram.

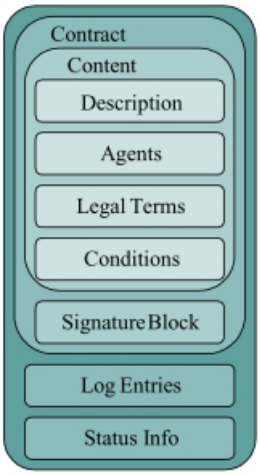

Figura 2.5: Representação da estrutura da mensagem do SeCO (Fonte: Greunz, Schopp, e Haes, 2001)

A implementação da estrutura do contentor SeCO (ver Figura 2.5) utiliza as normas *Document Type Definition* (DTD) [71] e XML. Tem pelo menos uma versão do contrato que está dividida num bloco de assinatura, para garantir que os dados não foram manipulados, e uma secção de conteúdo. É na secção de conteúdo que todos os dados relevantes para o contrato se encontram. Tem um bloco que descreve o produto (*product description*), um bloco com a informação sobre os parceiros do contrato (*agents*), um bloco para a descrição dos termos do contrato (*conditions*) e a informação legal respeitante à transação (*legal terms*). Para além da secção do contrato existem outras duas secções: a secção *status*, que mantém a informação sobre o estado do contrato; e a secção *log*, que regista os eventos que ocorrem durante o processo. A secção *log*, juntamente com a possibilidade de manter em cada contentor informação sobre os diversos contratos, permite manter a história negocial, sendo válido apenas o contrato mais recente. Em suma, o contentor SeCO pode ser visto como um objeto de informação onde estão contidos todos os dados que descrevem a lógica do fluxo de trabalho necessária à conclusão do contrato, os dados que descrevem os termos acordados e toda a informação necessária para os protocolos de comunicação.

Esta plataforma não permite *workflows* construídos dinamicamente. Relativamente à segurança, os autores abordaram: autenticação, autorização, integridade, rastreabilidade e não-repúdio. Para a autenticação, a arquitetura utiliza uma PKI, pelo que, a autenticação é garantida pela autoridade certificadora; a utilização da PKI juntamente com os direitos de acesso e identificação contidos no bloco *agents* da secção conteúdo permitem determinar o

acesso à informação, verificando a autorização; A integridade é assegurada uma vez que as secções do contrato são assinadas digitalmente pelos seus autores, o que permite a identificação do conteúdo que foi adulterado; Rastreabilidade é respeitada, uma vez que, cada contentor SeCO contém uma versão do contrato, as diferentes versões do contrato e a sua informação são enviadas entre as partes em todas as iterações até se obter o contrato final. A secção de *log* mantém o registo dos eventos que ocorrem durante o processo. A secção *content data* é digitalmente assinada. O bloco *agents* contém a informação sobre os parceiros envolvidos no processo impedindo o repúdio da ação por parte dos intervenientes.

2.4.4 Dias e Rafael

A plataforma proposta por Dias e Rafael [20, 21] é um resultado de trabalho académico com a existência de um protótipo, segue o paradigma SOA, é suportada por normas abertas (HTTP, XML e SOAP) e baseia-se em duas componentes: entidades e repositórios de serviços. De forma sucinta, para Dias e Rafael, uma entidade é um conceito abstrato que inclui departamentos e institutos que prestem serviços aos utilizadores de forma direta ou indireta [21, p.83]. Desta forma, uma entidade pode receber pedidos de serviços de utilizadores e providenciar esses serviços, ou, se não for capaz de o fazer, solicitar a outras entidades que forneçam o serviço. O repositório de serviços é um conceito baseado na proposta elaborada pelo projeto eGOV [70]. Os repositórios contém a descrição dos serviços, que são fornecidos pelas entidades. As entidades podem publicar a descrição dos seus serviços em diversos repositórios de serviços.

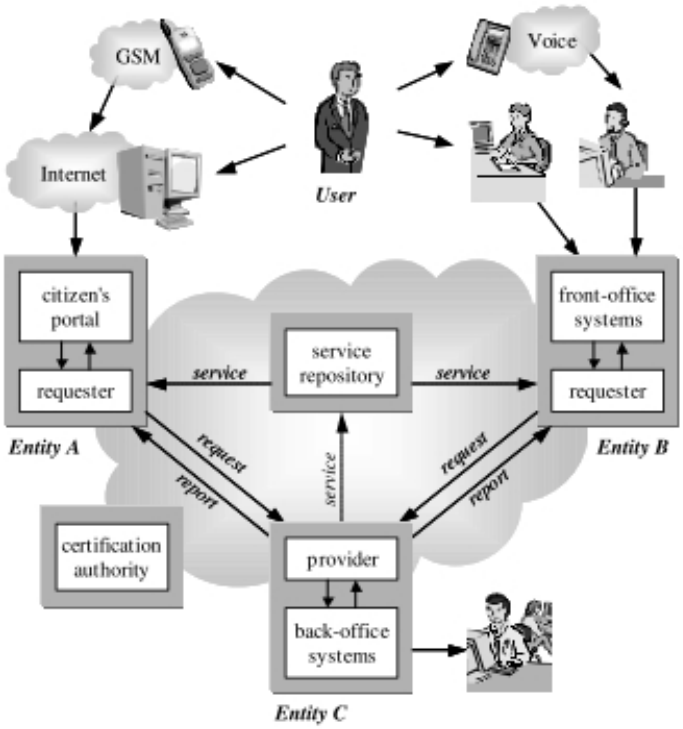

Figura 2.6: Arquitetura genérica (Fonte: Dias e Rafael, 2007)

As entidades comunicam através dos serviços, fazendo *upload* e *download* de descrições para os repositórios de serviços e enviando pedidos e relatórios entre si (ver Figura 2.6).

Para pedidos e relatórios a plataforma usa contentores baseados no proposto pelo projeto SeCO [59]. O contentor pedido é composto por dois blocos: bloco conteúdo e bloco assinatura. O bloco assinatura contém a assinatura digital e o certificado digital do assinante. O bloco conteúdo é composto pela secção detalhe, uma secção com lista de formulários e uma secção com a lista de anexos. A secção lista de formulários contém os formulários necessários para requisitar o serviço. A secção lista de anexos contém todos os documentos anexados. A secção de detalhes contém informação sobre o utilizador, a entidade e o pedido. O contentor relatório tem uma estrutura similar ao contentor pedido. A diferença encontra-se no bloco conteúdo que é composto pela secção detalhes, a secção *log* e a secção lista de resultados. O pedido que dá origem ao relatório é identificado na secção detalhes. A secção *log* é utilizada para descrever o estado anterior e atual do pedido. A secção resultado contém a lista dos resultados disponíveis.

Esta plataforma prevê *workflows* construídos dinamicamente. Quanto à segurança, com exceção da privacidade dos resultados e do controlo da participação no *workflow* a arquitetura aborda: autenticação; autorização; confidencialidade; integridade; rastreabilidade; e, não-repúdio. A autenticação dos utilizadores é feita pelas entidades. Quando necessário, o pedido contém a autenticação do utilizador. As entidades autenticam-se utilizando para tal, uma PKI. Relativamente à autorização, os pedidos, quando aplicável, contêm a informação sobre a autorização dos utilizadores. Esta informação é utilizada pelas entidades que recebem o pedido para permitir ou negar o acesso aos recursos. As entidades são classificadas de acordo com uma ou mais classes de autorização. Esta informação é incluída no certificado das entidades. As entidades que processam o pedido devem atribuir ou negar o acesso aos recursos de acordo com esta informação. A utilização do protocolo SSL durante a transferência dos contentores garante a confidencialidade da informação. O conteúdo é assinado digitalmente permitindo a identificação da informação alterada. A rastreabilidade é assegurada pelas secções *log* e lista de resultados do contentor relatório e pelo conteúdo assinado digitalmente nos contentores pedido e relatório. O não-repúdio é assegurado pela assinatura digital dos contentores.

2.4.5 *eMayor Project*

O projeto *eMayor* [35] foi um projeto financiado pelo 6º Programa Quadro de I&D da UE. O projeto *eMayor* pretendeu disponibilizar *Web Services* seguros e acessíveis para os pequenos e médios organismos da Europa. A plataforma *eMayor* segue o paradigma SOA.

A arquitetura *eMayor* (ver Figura 2.7) baseia-se em cinco grupos de serviços: *Core Web Services*; *User Interfaces*; *Security Services*; *Legacy Applications Support*; e, *Web Services Management*. Os *Core Web Services* constituem o núcleo central da plataforma, disponibilizam os *Web Services* que implementam vários processos governamentais, e gerem os processos intrinsecos da arquitetura. A infraestrutura UDDI/WDSL é utilizada para publicar os *Web Services*.

Os *User Interfaces* permitem que o utilizador final interaja com os *Web Services* de forma segura, a partir da Web ou de um dispositivo móvel. Os *Security Web Services* permitem que haja comunicação entre os *Web Services* e com todas as entidades através de políticas de segurança relevantes. Utilizam diversas normas para o efeito e com diversos fins: PKI; XML *Encryption*, XML *Signature*; *XML Key Management Specification* (XKMS); WS *Security*; SAML; e, *eXtensible Access Control Markup Language* (XACML).

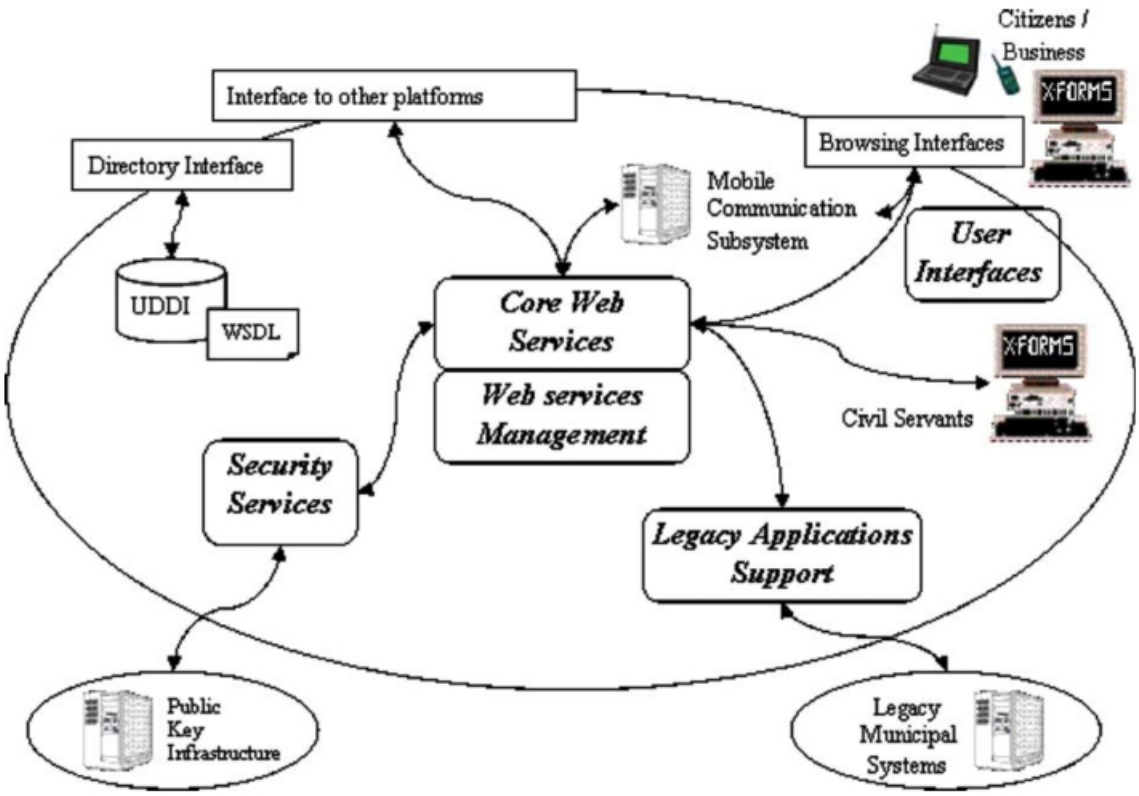

Figura 2.7: Arquitetura eMayor (Fonte: Kaliontzoglou et al., 2005)

Os *Legacy Applications Support* permitem que os sistemas já existentes sejam integrados. O *Web Services Management* faculta todas as ferramentas necessárias para a gestão da plataforma: Ferramentas para a gestão da segurança; e ferramentas que permitem adicionar, remover e configurar *Web Services*.

O *workflow* para a prestação de um serviço é pré-construído na plataforma *eMayor*, pelo que a plataforma não suporta a construção dinâmica de *workflows*. Relativamente à segurança, a PKI é utilizada para abordar a autenticação. A autorização é concretizada através da utilização de SAML e XACML. A utilização de assinatura digital, e em particular do XML-*DSIG*, do *time-stamp*, das funções criptográficas e do XML *Encryption*, permitem abordar a integração, a confidencialidade e o não-repúdio.

2.4.6 Web Digital Government

O Web Digital Government (WebDG) [9, 10, 44] é um sistema de gestão de Serviços Web (*Web Services Management System* (WSMS)) que se baseia no paradigma SOA, que utiliza Web Services: *Web Service Description Language* (WSDL), *Universal Description Discovery and Integration* (UDDI) e SOAP. A segunda versão[2] da implementação do protótipo da arquitetura foi disponibilizada em 2002. Este projeto junta investigadores da Virginia Tech

[2]http://europa.nvc.vt.edu/~dgov (project homepage);
http://europa.nvc.cs.vt.edu/~dgov/hongmei/ (prototype web site)

e da Universidade de Purdue com a *Family and Social Services Administration* (FSSA) de Indiana. O objetivo foi desenvolver uma plataforma de integração que permita o aumento da qualidade na prestação dos serviços por parte da FSSA.

O objetivo principal da plataforma é integrar os serviços utilizados pelo FSSA enquanto serviço da comunidade e, durante o processo, preservar a privacidade dos seus clientes. WebDG centra-se na composição de serviços e preservação da privacidade. Está organizado em três secções, cada uma correspondendo a um tipo de participante: *provider*, *registry* e *consumer*. O *provider* é quem entrega o serviço. *Registry* é onde o serviço é publicado. Finalmente, o *consumer* pode ser um de dois tipos: cidadão ou funcionário (*case officer*).

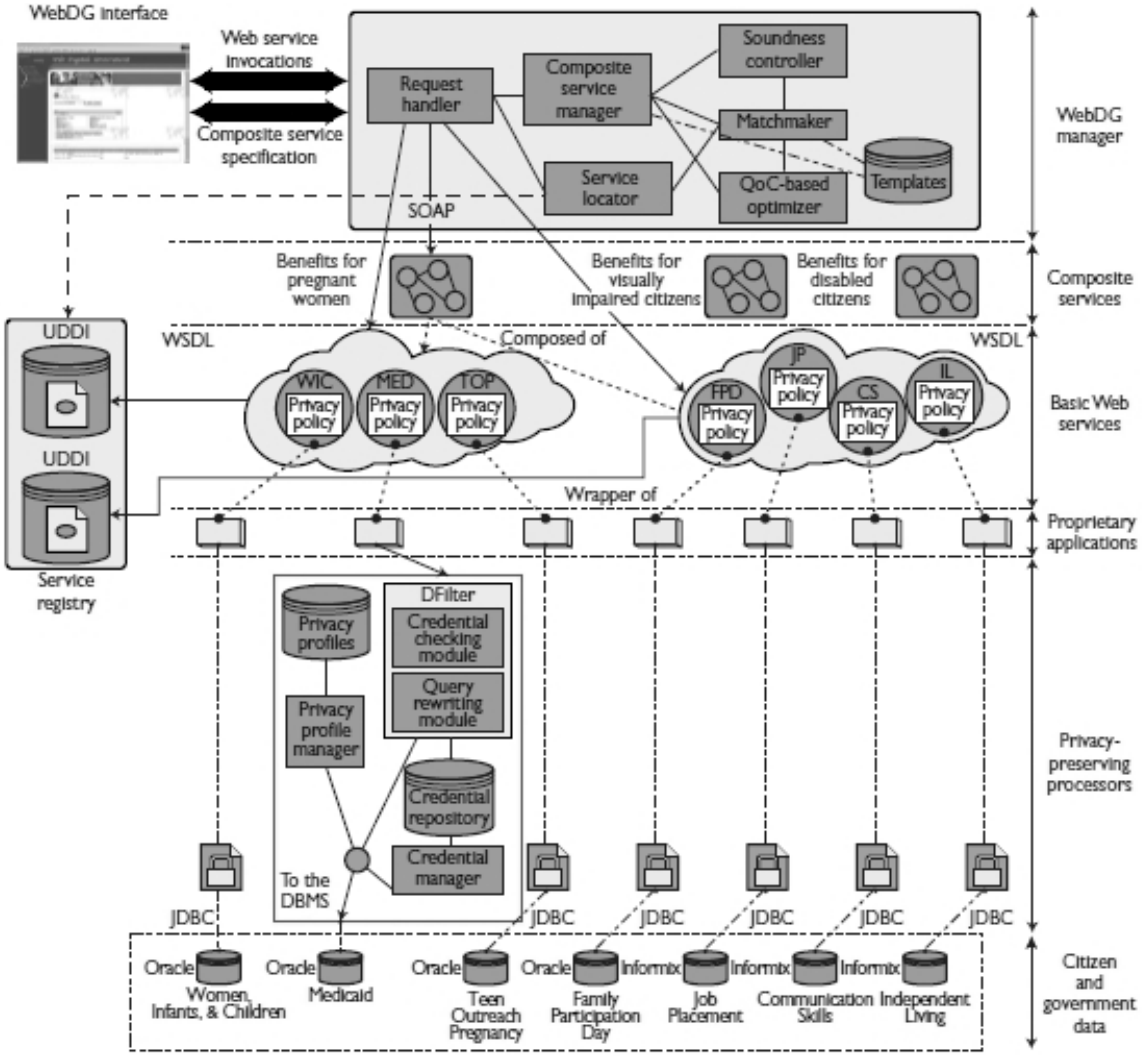

Figura 2.8: Plataforma Web Digital Government (Fonte: Medjahed et al., 2003)

A composição de serviços é usualmente criada pelo funcionário. É essencialmente uma lista de operações a ser executada. A plataforma, em seguida, gera um plano com base nas regras de composição da lista de serviços e como estes serviços interagem entre si. O plano de composição é submetido para composição semântica e sintática e o valor adicional dos diversos

planos gerados é avaliado.

Os autores afirmam que a privacidade é uma das principais preocupações que deve ser abordada pelo governo eletrónico. Para resolver esta questão os autores definiram três entidades típicas de governo eletrónico: utilizadores, serviços e bases de dados. Os utilizadores são aqueles que podem providenciar informação sensível. O acesso a esta informação depende de quem tenta aceder aos dados. Cada utilizador tem o seu próprio perfil de privacidade. Estes perfis de privacidade são dinâmicos.

Os *workflows* não são construídos dinamicamente. Relativamente à segurança, a autenticação é controlada na plataforma recorrendo a um par nome/senha; O acesso à informação é controlado pela aplicação e pelos perfis de privacidade criados e geridos pelos donos da informação - os utilizadores. Estes perfis de privacidade podem ser sobrepostos por algumas entidades governamentais (esta sobreposição é imposta por lei), atingindo a autorização;

2.4.7 Plataforma de Integração

A PI [1, 3, 4] é a plataforma de integração do governo português, utiliza o paradigma SOA. Inicialmente o projeto foi denominado por Framework de Serviços Comuns (FSC). Estão, neste momento, a ser adicionados novos serviços à plataforma, tendo 9 entidades aderido à utilização da plataforma (Abril de 2013).

A PI visa a integração gradual de serviços que os diversos organismos da Administração Pública disponibilizam para o público em geral. O desenvolvimento inicial desta plataforma ficou a cargo da UMIC – Agência para a Sociedade do Conhecimento[3]. No dia 1 de Maio de 2007, esta responsabilidade passou para a AMA – Agência para a Modernização Administrativa[4].

A Figura 2.9 representa o modelo para a integração entre organizações recorrendo à plataforma de integração. Esta plataforma baseia-se em normas abertas e permite a comunicação entre sistemas de distintos organismos governamentais. A plataforma utiliza Serviços Web e as normas XML, SOAP, *HyperText Transfer Protocol* (HTTP) [26], WSDL, *Web Services Security* (WSS) [47], *Web Services Addressing* (WSA) [74], *Web Services Reliable Messaging* (WSRM) [50] na comunicação entre os sistemas e para assegurar um conjunto de requisitos de segurança durante a comunicação [2], nomeadamente: confidencialidade, integridade, autenticidade e não-repúdio.

A plataforma de integração (ver figura 2.9) fornece um conjunto de funcionalidades para gerir e administrar a plataforma, para garantir a segurança e gerir e integrar serviços. A interface Web disponibiliza acesso à administração e gestão da plataforma. A gestão das interfaces dos *Web Services* é efetuada de acordo com as configurações realizadas pelos organismos. O motor de integração faz a conversão entre o modelo de dados dos organismos e o modelo de dados canónico. A segurança é garantida pelo módulo de Segurança.

A arquitetura não prevê *workflows* construídos dinamicamente. Em relação à segurança, a arquitetura tem mecanismos que permitem abordar a autenticação, autorização, confidencialidade, integridade, rastreabilidade e não-repúdio.

Para aceder aos serviços os clientes devem autenticar-se. Isto pode ser feito de duas formas: de uma forma mais tradicional, utilizando um par nome/senha; ou utilizando o Cartão de Cidadão[5]. Para garantir a autenticação em diferentes sistemas é utilizado o módulo Federação

[3]http://www.umic.pt/index.php
[4]http://www.ama.pt/
[5]http://www.cartaodecidadao.pt/

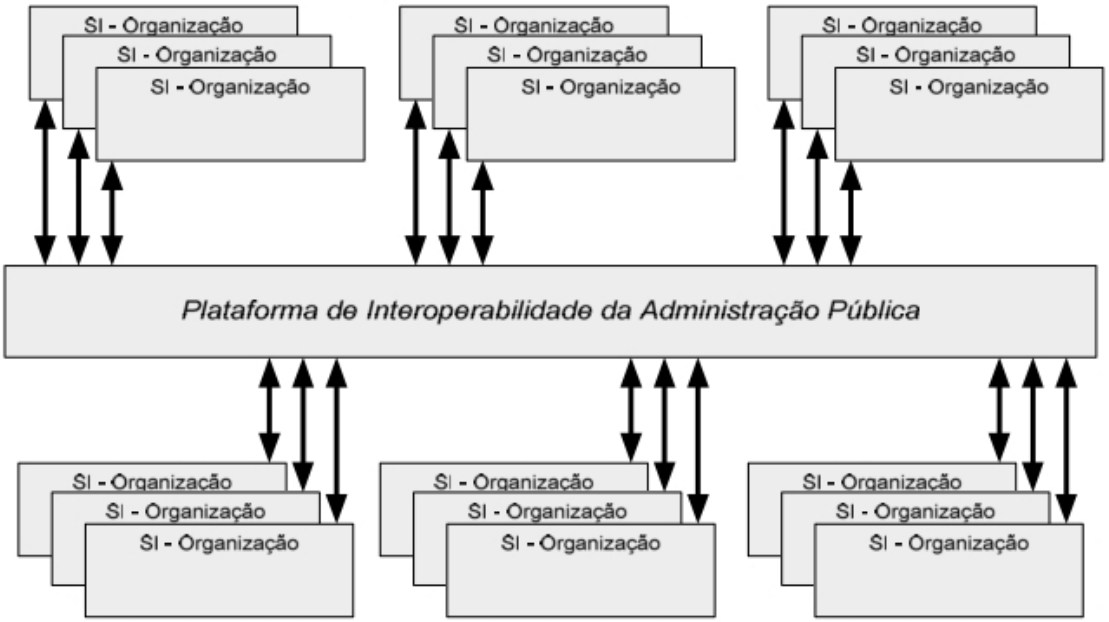

Figura 2.9: Modelo de referência para a integração (Fonte: Agência para a Modernização Administrativa, 2009)

de identidades, que permite o mapeamento de identidades entre as diferentes agências da Administração Pública.

O SAML *Token Profile* é utilizado para abordar a autorização. A arquitetura recorre à norma SAML *Token Profile* para a troca de autorizações. Durante a comunicação é utilizada a norma SSL para garantir a confidencialidade. A confidencialidade e a integridade do conteúdo da mensagem é assegurada com a utilização da WSS.

A plataforma utiliza um módulo denominado *Audit and Accounting* que regista os eventos que ocorrem na plataforma, abordando a rastreabilidade. As mensagens são assinadas digitalmente, os intervenientes no processo não podem negar a sua participação, assegurando o não-repúdio.

2.4.8 Arquitetura AIPA

A arquitetura AIPA [6] é uma arquitetura SOA. Foi uma arquitetura estabelecida no âmbito dos programas financiados pela autoridade italiana para as tecnologias da informação na Administração Pública (*Autorità per l'Informatica nella Pubblica Amninistrazion*). A arquitetura é composta (ver Figura 2.11) por: *Web Servers*; *Application Servers*. Os *Web Servers* permitem que os utilizadores solicitem um serviço. Os *Application Servers* recebem o pedido encaminhado pelo *Web Server* e redireccionam-no para o motor *servlet* que, por sua vez, constrói um cabeçalho apropriado e reencaminha-o para o *dispatcher*. Este cabeçalho contém informação sobre o solicitador, os privilégios de acesso e o serviço. Esta informação é utilizada pelo *dispatcher* para identificar o utilizador e os seus privilégios de acesso relativamente ao pedido solicitado, ativar as medidas de segurança adequadas, verificar a correção e completude do pedido de serviço e é responsável por identificar para qual dos prestadores de serviço deve

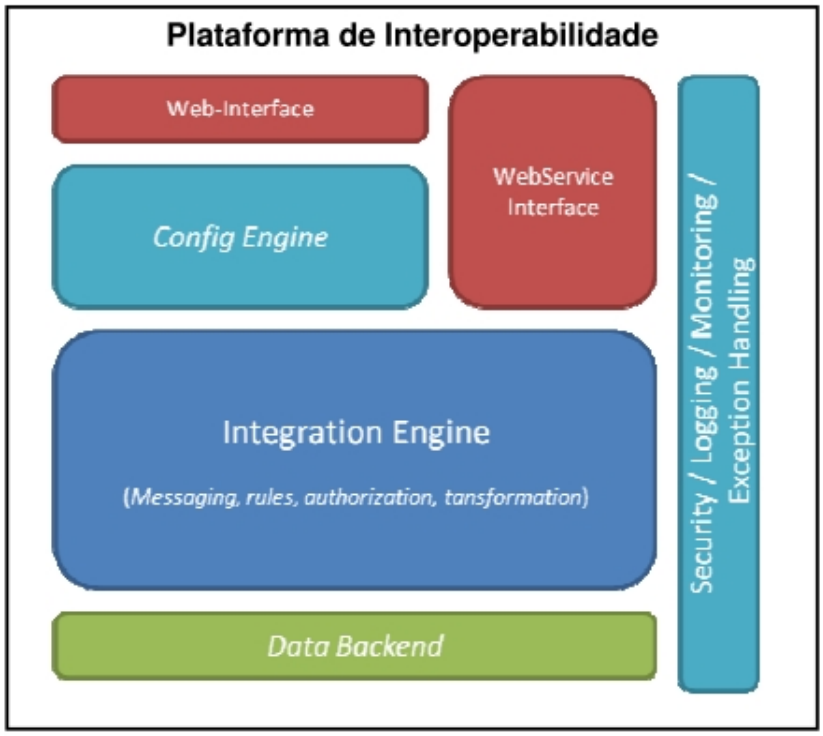

Figura 2.10: Plataforma de Interoperabilidade (Fonte: Agência para a Modernização Administrativa, 2009)

enviar os sub-pedidos ou invocar os sistemas locais através do gestor de informação. Por sua vez, o gestor de informação é capaz de aceder aos serviços de gestão interna de informação.

Esta arquitetura é capaz de orquestrar serviços dinamicamente. Isto é conseguido pelo *dispatcher*. Relativamente à segurança, nas fontes obtidas, apenas a autorização e autenticação foram abordadas. Isto é conseguido no motor *servlet* quando constrói o cabeçalho e no *dispatcher* que é responsável por examinar cabeçalho.

2.4.9 Access-eGOV

O Access-eGOV[6] [22, 38, 39] foi um projeto financiado pela Comissão Europeia, número de referência FP6-2004-27020, sob o 6º Programa Quadro das Tecnologias para a Sociedade da Informação (IST); teve a duração de 36 meses (1 de Janeiro de 2006 a 31 de Dezembro de 2008). O principal objetivo do projeto foi desenvolver uma plataforma de integração baseada em Tecnologias Semantic Web, através da utilização das normas *Web Service Modeling Ontology* (WSMO), *Web Service Modeling Language* (WSML) e a implementação *Web Service Modelling Execution Environment* (WSMX), e em arquiteturas distribuídas, utilizando ligações P2P. Foram desenvolvidas três aplicações piloto, uma para cada país envolvido no projeto: Alemanha, Eslováquia e Polónia. Cada teste piloto tem um cenário diferente: casar (Alemanha), obtenção de uma licença de construção (Eslováquia), e criação de uma empresa

[6]http://www.access-egov.org

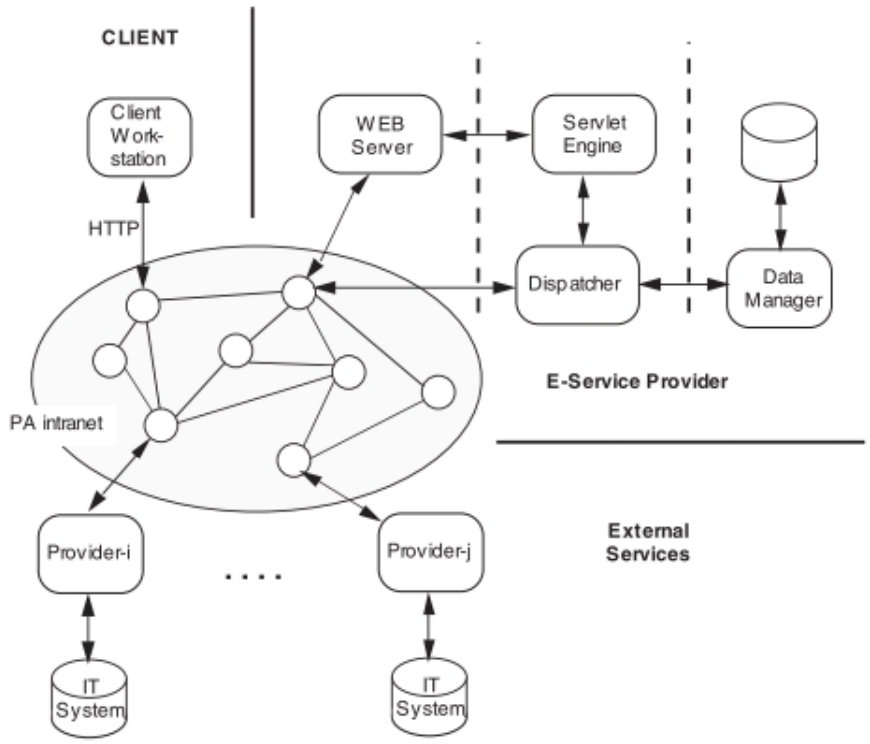

Figura 2.11: Arquitetura do Sistema Access-eGov. (Fonte: Arcieri et al., 2002)

(Polónia). A arquitetura Access-eGOV divide-se em três grupos: *Infrastruture*; *Personnal Assistant Client*; e *Public Administration tools* (ver Figura 2.12).

O grupo principal da arquitetura é o *Infrastruture*. É aqui que estão localizadas as componentes que permitem a integração e composição dos serviços. Este grupo está subdividido em componentes categorizadas: *Data repositories*; *Core components*; *Connection manager*; *Security component*; *Dynamic components* (módulos de procura, composição e execução) e *Mediator*. A componente *Data repositories* armazena a descrição semântica dos serviços, cenários, ontologias e objetivos dos utilizadores. As bases de dados distribuídas, que a compõem, estão interligadas e são regularmente atualizadas. A principal funcionalidade das *Core components* é permitir que os utilizadores executem operações sobre os *Data repositories*. O *Connection manager* está também localizado na componente central, a sua função é permitir a comunicação dentro do sistema e entre o sistema e o seu exterior. A componente *Security* é transversal à arquitetura. No entanto, encontra-se incorporada no grupo *Infrastructure*.

Devemos mencionar o módulo *Mediation*. Este módulo é utilizado como um mediador entre o *Personal Assistant Client* e o restante sistema no acesso aos *Data repositories* (ver Figura 2.13). Pesquisa e executa os serviços desejados utilizando os módulos *discovery*, *composition* e *execution* incluídos no *Dynamic Components*.

O *Personal Assistant Client* é a componente que tem como principal objetivo guiar e ajudar os utilizadores a encontrar os serviços desejados.

Finalmente, as *Public Administration tools* integram a componente que é usada pelas agências governamentais para publicar os seus serviços. Por outras palavras, estas são as

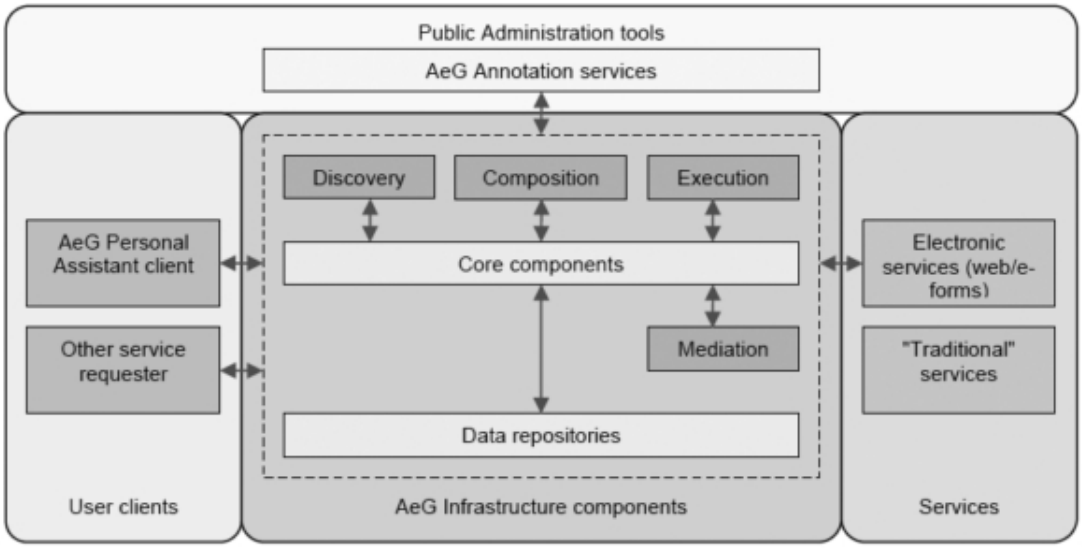

Figura 2.12: Arquitetura do Sistema Access-eGov. (Fonte: Durbeck, Schillinger e Kolter, 2007)

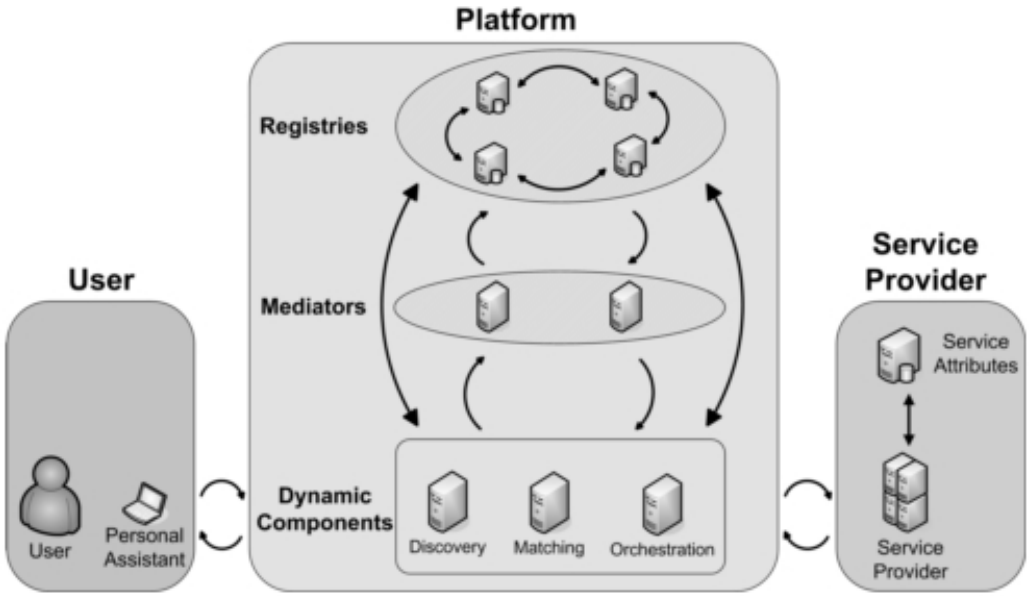

Figura 2.13: Vista estrutural da arquitetura do Access-eGov. (Fonte: Kolter, Schillinger e Pernul, 2007)

ferramentas que a administração pública utiliza para criar as entradas dos serviços nos repositórios de dados, tornando os serviços acessíveis aos utilizadores. Os serviços fazem parte das agências que os oferecem, não pertencendo à arquitetura.

A arquitetura Access-eGov permite *workflows* pré-construídos. A arquitetura foi desenvol-

vida para utilizar PKI, que permite a identificação do utilizador e a sua autenticação. Pode igualmente utilizar outros métodos de autenticação.

Para os autores do Access-eGOV a autorização é um dos requisitos de segurança mais relevantes, e optaram pela utilização de um sistema *Attribute-based Access Control* (ABAC) como solução. Para implementar o ABAC foi utilizado o XACML. Para além da descrição semântica funcional, isto é feito adicionando uma anotação ao serviço, que contém os seus atributos de segurança. Quando o serviço é pedido, é tomada em consideração a validade das credenciais e a informação contida na anotação.

As mensagens utilizadas pelo sistema têm uma estrutura baseada em XML. XML *signature* e XML *encryption* são utilizados nas mensagens e o protocolo SSL é utilizado na rede, garantindo a confidencialidade. Integridade, a assinatura XML é utilizada. Isto garante a deteção das mensagens que foram alvo de alteração. A arquitetura confia em diversas funcionalidades de registo espalhadas pelos seus componentes, o que garante a rastreabilidade. O não-repúdio é assegurado, para além da capacidade de registo, o sistema confia no conteúdo das mensagens assinado digitalmente para prevenir o seu repúdio.

2.4.10 Análise

Não é o nosso propósito analisar todos os aspetos técnicos das soluções propostas relativamente à integração. De facto, o âmbito da nossa análise restringe-se aos pontos identificados no início da secção 2.4. O objetivo desta análise é determinar quais as arquiteturas de integração que permitem que *workflows* sejam construídos dinamicamente e se têm mecanismos para assegurar a privacidade dos resultados. A Tabela 2.1 contém a compilação dos resultados obtidos pela análise individual a cada uma das arquiteturas. Nesta tabela, encontram-se igualmente os requisitos de segurança que cada uma das arquiteturas e/ou plataformas concretizou.

Tabela 2.1: Quadro agregador da análise efetuada às arquiteturas de integração.

	eGov	OSCI	D&R	SeCO	eMayor	WebDG	IP	AIPA	Access
Exec. din.	✗	✗	✓	✗	✗	✗	✗	✓	✗
Autenticação	✓	✓	✓	✓	✓	✓	✓	✓	✓
Autorização		✓	✓	✓	✓	✓	✓	✓	✓
Confiden.	✓	✓	✓		✓	✗	✓		✓
Integridade	✓	✓	✓	✓	✓	✗	✓		✓
Não repúdio	✓	✓	✓	✓	✓	✗	✓		✓
Rastreabi.		✓	✓	✓					✓
Priv. dos res.			✗					✗	
C. Part. *work.*			✗					✗	

Legenda:
✓ verifica
✗ não verifica
não se obteve informação ou não é aplicável

Duas das arquiteturas e/ou plataformas estudadas permitem *workflows* construídos dinamicamente (D&R e AIPA). No entanto, ambas não contemplam mecanismos que permitam lidar com a privacidade dos resultados nem com o controlo da participação no *workflow*.

Podemos concluir, por isso, que nenhuma das arquiteturas e/ou plataformas estudadas aborda as condições definidas: (1) ser capaz de construir *workflows* dinamicamente; (2) proteger os resultados de forma adequada mesmo quando o destinatário não é conhecido no momento de criação do resultado; (3) permitir que o participante decida sobre a sua participação ou não no *workflow*.

2.5 Modelos de controlo de acesso

Nesta secção temos como objetivo apresentar os modelos de controlo de acesso que são adequados para facultar ou negar o acesso a serviços ou resultados. Todos os modelos de controlo de acesso que apresentamos podem ser utilizados na nossa contribuição.

2.5.1 DAC

O modelo de controlo de acesso DAC foi definido em [65] pelo Departamento de Defesa dos Estados Unidos da América. Este modelo baseia-se na identidade para permitir ou negar o acesso a um objeto. Tal como o nome indica, permite que o dono do objeto, ou alguém com a autorização necessária, defina quem possa ter acesso ao objeto e com que direitos. Os direitos de acesso quando o modelo DAC é utilizado podem mudar de forma dinâmica. Na prática, quem tem autorização para controlar o acesso a um objeto, poderá remover os direitos de acesso a outros, alterá-los ou atribuir direitos de acesso em qualquer momento.

2.5.2 *Chinese Wall*

O modelo de controlo de acesso *Chinese Wall* foi definido, por Brewer e Nash, em [11]. Este modelo de controlo de acesso visa essencialmente os conflitos de interesse, embora também aborde a confidencialidade e a integridade. Os conflitos de interesse existem quando uma pessoa pode obter informação ou estar na posse de informação sensível sobre pessoas, produtos ou organismos concorrentes.

O modelo de controlo de acesso é suportado por três níveis de abstração:

- Objetos - encontram-se no nível mais baixo e cada objeto deve conter informação relativo a um único organismo (e.g. ficheiro);
- Conjunto de dados de um organismo - encontram-se no nível intermédio de abstração. Neste nível são agrupados todos os objetos pertencentes a um mesmo organismo;
- Classes de conflitos - no nível mais alto de abstração são agrupados todos os conjuntos de dados de organismos que são concorrentes.

O modelo é relativamente simples. Inicialmente, quando a pessoa não tem informação sobre nenhum dos organismos que estão agrupados dentro de uma das classes de conflitos, poderá obter informação sobre qualquer um desses organismos. Ou seja, aceder a qualquer um dos objetos que estejam associados a qualquer dos organismos numa mesma classe de conflito. A partir do momento em que a pessoa acede a algum dos objetos de um dado organismo, deixa de ter a possibilidade de aceder a objetos de organismos que estejam na mesma classe de conflito do organismo a que o objeto pertence. No entanto, poderá continuar a aceder aos objetos que estejam no conjunto de dados pertencente ao organismo. Estas restrições apenas

se aplicam dentro de uma classe de conflitos. A pessoa poderá aceder a objetos que pertençam ao mesmo conjunto de dados de um organismo desde que os organismos pertençam a classes de conflito diferentes.

2.5.3 Clark-Wilson

O Modelo Clark-Wilson aborda a integridade e tem como principal motivação as necessidades do setor empresarial [17]. Neste caso em particular, e não desprezando a confidencialidade da informação a que é dada importância noutros modelos, o enfoque recai sobre a integridade dos dados (principalmente para evitar fraude e erros). Este modelo baseia-se no conceito de transação bem formada (*well-formed transaction* (WFT)) que consiste na manipulação correta de dados por parte de um utilizador de forma a assegurar a integridade desses mesmos dados.

O modelo é composto por um conjunto de elementos básicos:

- *Constrained Data Item* (CDI) - dados sujeitos a restrições de integridade;
- *Unconstrained Data Item* (UDI) - dados não sujeitos a restrições de integridade;
- *Integrity Verification Procedures* (IVP) - têm como objetivo confirmar que todos os CDI presentes no sistema verificam as especificações de integridade no momento em que é executado;
- *Transformation Procedures* (TP) - tem como função passar os CDI de um estado válido para outro.

Clark e Wilson utilizam tripletos do tipo <utilizadorID, TP, {CDI, CDI, ...}> para associar a TP aos dados (CDI) e à identificação do utilizador que está autorizado a aceder aos dados através de uma TP. A TP quando aplicada a um UDI tranforma o UDI num CDI.

O modelo consiste em dois conjuntos de regras: regras de certificação (*Certification*) e regras de aplicação (*Enforcement*). As regras de aplicação asseguram que os dados mantêm a integridade e garantem o acesso autorizado aos CDI e às TP e não carecem de intervenção externa. As regras de certificação são, na sua generalidade, mais complexas e requerem intervenção externa. Estes conjuntos de regras garantem a integridade interna e externa dos dados.

2.5.4 RBAC

O RBAC foi proposto por Ferraiolo e Kuhn em [25]. Tem como objetivo limitar o acesso a utilizadores autorizados baseando-se nas funções ou papéis que o utilizador desempenha. As autorizações são atribuídas ao papel e não ao utilizador.

O modelo RBAC baseia-se em três regras:

- Atribuição de função - o sujeito pode aceder a um objeto se lhe tiver sido atribuído uma função (*role*);
- Autorização da função - o sujeito apenas pode ter funções para as quais está autorizado;
- Autorização de permissões - o sujeito só poderá utilizar uma permissão se esta estiver autorizada para a função atribuída ao sujeito.

Este modelo permite que a um sujeito possam ser atribuídas várias funções e que uma mesma função possa ser atribuída a vários sujeitos. Tal como acontece com a relação entre sujeitos e funções, a relação entre funções e permissões e a relação entre permissões e operações a que estão associadas têm uma cardinalidade de muitos para muitos.

2.6 Conclusões

Neste capítulo começámos por definir um conjunto de conceitos com o objetivo de uniformizar a sua utilização e interpretação ao longo do documento. Seguiu-se a identificação de um conjunto de quadros de referência para a interoperabilidade de forma a identificar as abordagens que sugerem e quais os quadros de referência relevantes para a nossa contribuição. Identificámos igualmente paradigmas relevantes para a integração de serviços e fizémos um levantamento, identificámos e descrevemos um conjunto de arquiteturas e/ou plataformas de integração. A análise teve como objetivo determinar as arquiteturas e/ou plataformas que constróiem *workflows* dinamicamente. E destas, aquelas que têm mecanismos para manter os resultados privados. Concluímos o capítulo com a descrição de um conjunto de modelos de controlo de acesso com o objetivo de indicar modelos que podem ser utilizados para a atribuir ou negar o acesso a serviços e resultados.

Da análise efetuada na secção 2.4.10 podemos concluir que apenas duas das arquiteturas e/ou plataformas permitem a construção de *workflows* dinâmicos (D&R e AIPA). No entanto, em nenhum dos casos foi abordada a privacidade dos resultados nem o controlo da participação no *workflow*.

Da análise efetuada pudemos também constatar que a tecnologia PKI é uma das soluções para a maioria dos requisitos de segurança identificados pelas arquiteturas e/ou plataformas estudadas. Esta escolha tem consequências na estrutura das arquiteturas. A utilização de assinaturas digitais está prevista na lei e está a tornar-se bastante difundida, e.g. a maioria dos países da UE estão a transformar o seu cartão de identificação em cartões eID que agregam diversa informação sobre o utilizador, a chave privada, a chave pública e os certificados correspondentes.

Relativamente aos modelos de controlo de acesso descritos, os objetivos de cada um deles é diferente. O DAC e o RBAC são aqueles que têm objetivos mais próximos no entanto, a sua abordagem é distinta. Enquanto que o DAC atribui as permissões de acesso a um utilizador ou a um grupo, no modelo RBAC essas permissões são atribuídas a uma função (role). No entanto para que esta função seja atribuída a um utilizador (sujeito) existe um conjunto de regras que deve ser respeitado.

Relativamente aos modelos *Chinese Wall* e Clark-Wilson ambos divergem no objetivo principal e na abordagem que seguem. O modelo *Chinese Wall* visa o conflito de interesses impedindo o acesso a informação caso o utilizador já tenha acesso privilegiado a dados de organismos concorrentes. O modelo Clark-Wilson tem o seu enfoque na integridade dos dados. Para garantir a integridade foram definidas regras de certificação e de aplicação. É com base nestas regras e nos tripletes (que associam utilizadores, TP e CDI) que é feito o controlo de acesso e se garante a integridade dos dados.

Capítulo 3

Arquitetura proposta e modelo de segurança

No capítulo anterior identificámos os quadros de referência para a interoperabilidade relevantes para a nossa contribuição, nomeadamente o EIF da UE e o GIE de Portugal. Identificámos e descrevemos um conjunto de arquiteturas e/ou plataformas de integração e analisámos este mesmo conjunto com o intuito de determinar o mecanismo utilizado na prestação de serviços compostos e determinar se a privacidade dos resultados era garantida quando se tratasse da criação de *workflows* dinâmicos. Adicionalmente, procedemos ao levantamento dos mecanismos de segurança que estas arquiteturas apresentam. Embora não seja a tecnologia principal em todas as arquiteturas e/ou plataformas estudadas, os Serviços Web e os mecanismos de segurança a eles associados estão presentes na generalidade das arquiteturas e/ou plataformas.

Como concluímos (ver Secção 2.4.10), das arquiteturas de integração identificadas, a arquitetura Access-eGov pré-constrói os *workflows* para a prestação do serviço e apenas duas (D&R e AIPA) permitem a construção dinâmica de *workflows*. No entanto, as arquiteturas D&R e AIPA não contemplam a privacidade dos resultados. Também no capítulo anterior, identificámos os modelos de controlo de acesso relevantes para o trabalho.

Neste capítulo apresentamos uma arquitetura genérica para a construção de *workflows* dinâmicos de governo eletrónico de forma segura. Algumas das ideias apresentadas neste capítulo foram originalmente propostas por Dias e Rafael em [21]. Como demonstraremos nas secções 3.3.4 e 3.4 todos os objetivos identificados na secção 3.1 são atingidos. Esta arquitetura permite que os participantes na prestação do serviço, que designamos por Entidades, decidam se participam ou não, a quem vão delegar o *workflow* e a quem os resultados podem ser entregues. Todas as decisões têm suporte numa PKI e nos certificados com atributos que cada Entidade possui. A arquitetura segue os princípios-base publicados pelo Autor em [42].

O capítulo está organizado da seguinte forma: na secção 3.1 identificamos, descrevemos e justificamos os objetivos que pretendemos atingir; a secção 3.2 identificamos as questões de segurança associadas à execução de *workflows* dinâmicos tendo por base a composição de serviços prestados por diversas Entidades; a arquitetura é descrita na secção 3.3; Por fim, o capítulo é concluído na secção 3.4.

3.1 Objetivos

Nesta secção apresentamos o conjunto de objetivos que a arquitetura que propomos deve observar.

3.1.1 Integração de serviços

A AP é composta por diversos organismos que providenciam serviços dispersos. Como vimos, cidadãos e empresas desejam que os serviços sejam disponibilizados de acordo com as suas necessidades. A aglomeração de serviços pode seguir o paradigma *eventos da vida*, quando aplicável a serviços destinados aos cidadãos, ou *processos de negócio*, quando os serviços se destinam às empresas.

Como observámos na secção 2.2, os quadros de referência para a interoperabilidade que devem ser respeitados pela solução a propor são o EIF, quando aplicável ao contexto supranacional e entre os sistemas da União Europeia, e o GIE quando aplicável aos sistemas no contexto português. Ambos os quadros de referência para a interoperabilidade sugerem a integração de processos para disponibilizar serviços para os clientes da AP.

Este é um dos objetivos fundamentais para a arquitetura. Só assim será possível garantir que possam ser criados *workflows* que possibilitem a prestação de serviços complexos que visem a abordagem centrada no cliente.

3.1.2 Concorrência

A maior parte dos serviços prestados pela AP pode ter vários prestadores (e.g. serviços do governo local, serviços de saúde, serviços de educação). Qualquer arquitetura que vise a prestação de serviços deve ser capaz de refletir esta realidade. Além disso, existem várias entidades privadas que são legalmente capazes de prestar serviços concorrentes (e.g. firmas de notariado privadas, escolas privadas, clínicas de saúde privadas), que devem igualmente ser capazes de publicar os seus serviços. Para refletir esta multiplicidade de opções, a arquitetura deve ser capaz de determinar diferentes caminhos de forma a que os diferentes organismos da AP possam colaborar entre si para prestar um serviço. Ou seja, dois pedidos diferentes para o mesmo serviço podem resultar em diferentes caminhos de *workflow*, recorrendo a organismos diferentes, para atingir o resultado final mais satisfatório.

3.1.3 Versatilidade

A Administração Pública é um "organismo vivo" que está em constante mutação. Ao longo do tempo, novas divisões, leis e procedimentos são criados e outros são alterados ou removidos, o que influencia a disponibilidade de serviços que são prestados pela AP, removendo ou renovando serviços obsoletos ou criando novos. Ademais, a adoção de uma arquitetura de integração não é conseguida do dia para a noite, é um processo gradual que depende da complexidade e das restrições (técnicas, legais, sociais, etc.) dos serviços a disponibilizar. Porém, mais cedo ou mais tarde, todos os serviços devem ser adicionados e disponibilizados. De forma a representar o dinamismo do ambiente da AP e para assegurar a sua constante evolução, as arquiteturas de integração devem ser versáteis. Este objetivo pode ser atingido permitindo a adesão ou desvinculação de organismos públicos e/ou organismos privados às plataformas que respeitem a arquitetura, em qualquer altura, e permitindo a publicação, atualização ou remoção dos seus serviços sempre que necessário.

3.1.4 Escalabilidade

Uma arquitetura de integração para governo eletrónico deve estar preparada para que todos os organismos da Administração Pública estejam representados e disponibilizem os seus serviços. Ademais, a arquitetura poderá, inclusivamente, suportar a existência de serviços, privados ou semi-privados, prestados por organismos privados. A arquitetura, por estes motivos, não deve colocar problemas de escalabilidade diferentes dos que já se colocam com as soluções existentes.

3.1.5 Segurança

Na forma tradicional de prestação de serviços na Administração Pública, o cliente garante um conjunto de propriedades de segurança através da sua participação direta (ou através de um representante de confiança) no processo de prestação do serviço. Um exemplo é a entrega do registo criminal para acesso a um emprego. No caso da prestação tradicional do serviço, o cliente decide a quem quer entregar o documento. Isto não se verifica quando uma plataforma de integração é utilizada; o cliente pode não ter o poder de decidir a quem a informação deve ser enviada, uma vez que, na maior parte dos casos, o cidadão não é capaz de seguir os seus próprios dados. A arquitetura de integração deve providenciar um conjunto de mecanismos que garantam a prestação segura de serviços, com o objetivo de respeitar todos os direitos básicos de privacidade dos cidadãos. De facto, na pior das hipóteses, este tipo de soluções (arquiteturas que visam a prestação integrada de serviços) representa um perigo real para a privacidade do cliente e para a confidencialidade dos seus dados.

Uma arquitetura de integração envolve serviços, informação, dados e processos que são ou podem ser sensíveis. Na maioria dos países, estes dados estão protegidos por lei, restringindo-os ao dono da informação, o utilizador e a algumas agências bem definidas da AP. É então essencial que a arquitetura respeite um conjunto de propriedades para suportar uma interoperação correta e segura e que respeite o sistema legal dos países envolvidos.

Em [5, 13, 21, 34] os autores estabelecem um conjunto de propriedades de segurança para infraestruturas digitais de Governo Eletrónico. Em cada um dos artigos foi definido um conjunto diferente de propriedades, no entanto as propriedades base são essencialmente iguais. Este conjunto de propriedades é também mencionado na EIF, no GIE e na FEAF.

Baseando-nos nos trabalhos apresentados e nos quadro de referência para a interoperabilidade cuja nossa contribuição deve respeitar (EIF e GIE), propomos as seguintes propriedades de segurança:

- Autenticação – ser capaz de identificar os atores do sistema;
- Autorização – ser capaz de facilitar ou negar o acesso aos recursos do sistema (serviços, resultados, etc.);
- Confidencialidade – ser capaz de prevenir o acesso não autorizado à informação;
- Integridade – ser capaz de prevenir a corrupção ou adulteração da informação;
- Rastreabilidade – ser capaz de atribuir as ações efetuadas por um ator a ele próprio;
- Não-repúdio – ser capaz de prevenir que os intervenientes das transações neguem o seu envolvimento.

3.2 Problema

Para atingir o objetivo genérico de facultar *workflows* dinâmicos a orquestração dinâmica de serviços é fundamental. No entanto, esta abordagem cria um conjunto de problemas adicionais de segurança.

Como vimos na secção 2.1, a construção dinâmica de *workflows* consiste na definição da sequência de passos durante a execução do *workflow*. Isto significa que os participantes no *workflow* poderão não ser conhecidos *a priori*. Este pequeno detalhe é de extrema importância, obrigando a preocupações adicionais ao nível da segurança:

- Confiança nos participantes - um participante está acreditado para fornecer o serviço que publicou? É capaz de selecionar o próximo participante a participar na prestação do serviço?
- Preservação da privacidade - parte dos dados manipulados ou produzidos ao longo de um *workflow* poderá ser sensível. Como garantir que os participantes não acedem às partes desses dados de que não são destinatários?
- Conflitos de interesse - um participante poderá recusar-se a participar na prestação do serviço caso exista outro participante com o qual tenha conflitos de interesse?

Embora estas preocupações estejam de alguma forma embutidas nas características de segurança identificadas anteriormente, elas têm um impacto direto nas medidas de segurança que têm de ser previstas na arquitetura de integração que vamos apresentar.

3.3 Solução proposta

Esta secção tem como objetivo apresentar a arquitetura que propomos para atingir os objetivos definidos na secção 3.1 e abordar as questões enumeradas na secção 3.2. Começamos por apresentar as ideias gerais da arquitetura (ver secção 3.3.1); na secção 3.3.2 apresentamos o modelo de segurança; na secção 3.3.3 apresentamos as componentes da arquitetura; por fim, analisamos a arquitetura relativamente aos objetivos definidos na secção 3.3.4.

3.3.1 Ideias gerais

O objetivo desta secção é apresentar genericamente a arquitetura, permitindo uma rápida perceção da estrutura base da arquitetura e do seu funcionamento, quando implementada.

Princípios base

A ideia principal da arquitetura de integração de serviços (ver Figura 3.1) e o seu conceito base são relativamente simples: a arquitetura é composta por um conjunto de Entidades que representam organismos reais e que colaboram entre si para prestar serviços. As Entidades podem prestar serviços complexos através da composição de serviços mais simples disponibilizados por outras Entidades. Os serviços são publicados pelas Entidades que estão acreditadas para prestar o serviço, que os fornecem registando-os nos repositórios de serviços. As Entidades decidem o acesso de outras Entidades ao serviço de acordo com as credenciais de autorização das requerentes.

A arquitetura baseia-se em quatro pressupostos nucleares:

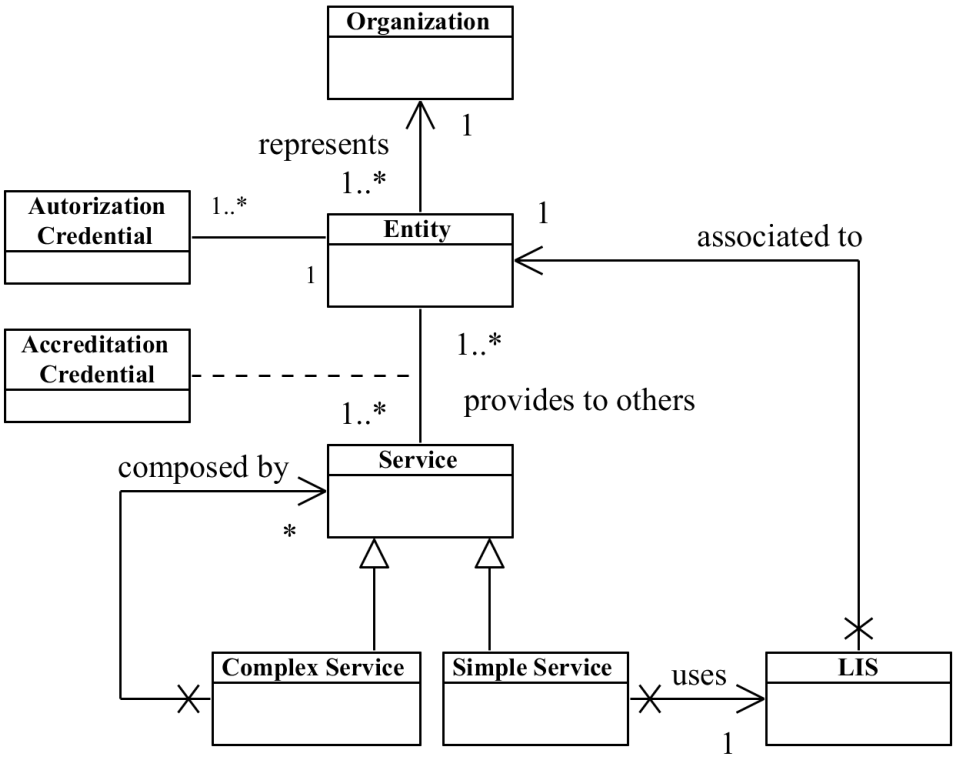

Figura 3.1: Diagrama UML de conceitos representativo dos conceitos de domínio e a forma como se relacionam.

- A integração de serviços necessária à prestação de serviços complexos é conseguida através da composição dos serviços que são prestados por Entidades. Os resultados produzidos por estes serviços complexos podem ser consumidos por outras Entidades.
- Todos os serviços são prestados por Entidades. Todas as Entidades podem prestar serviços complexos. No entanto, os serviços simples apenas podem ser prestados por Entidades que estejam associadas a um Sistema de Informação Local (SIL) de um Organismo (público ou privado).
- A gestão do processo de prestação de um serviço (*workflow*) não está centralizado em nenhuma Entidade, com a exceção do *workflow* de um serviço simples. Qualquer Entidade pode delegar a gestão do todo ou de partes do *workflow* a outras Entidades através da solicitação de um serviço. Isto significa que o processo de construção do *workflow* de um serviço complexo é dinâmico e que as Entidades que participam na prestação do serviço podem variar para pedidos semelhantes. Mais, e devemos destacar a extrema importância deste ponto, não há conhecimento prévio de quais as Entidades que participam no *workflow* de um serviço complexo.
- Os resultados produzidos durante o processo de prestação de um serviço são mantidos pelas Entidades que os produziram. Estes resultados são apenas disponibilizados às Entidades que necessitam deles quando as mesmas os solicitem explicitamente. Este comportamento é imposto fundamentalmente pelo objetivo de privacidade. É também

uma consequência direta do não conhecimento do destinatário final do resultado aquando da sua produção, uma vez que os *workflows* são dinâmicos. No entanto, as localizações onde os resultados podem ser obtidos fazem parte das mensagens que são trocadas entre Entidades, permitindo que os resultados possam ser solicitados quando os seus consumidores finais são atingidos.

Caso de uso ilustrativo

Para exemplificar o comportamento de uma concretização da arquitetura num cenário real, vamos utilizar um exemplo de assistência médica. Esta descrição pode ser acompanhada na Figura 3.2. Por simplicidade, a figura não contém todas as mensagens que são trocadas entre as Entidades envolvidas no processo. Embora simples, esta representação é suficiente para observarmos como a arquitetura abrange diversos Sistemas de Informação de diferentes agências da Administração Pública. Pedidos e respostas de carimbos temporais e da validação das certificações foram omitidas na figura. Mais, assumimos que todos os passos do processo da prestação do serviço funcionam sem problemas: todos os pedidos são satisfeitos; todas as Entidades e SIL estão acessíveis; e todas as validações de acreditação e de autorização são avaliadas positivamente. As mensagens trocadas entre as Entidades são de quatro tipos: Pedidos de serviço (*Request Service*) - utilizadas aquando da solicitação de um serviço; Notificação (*Notification*) - informam sobre a alteração de estado de um serviço, sobre o endereço de um resultado, ou da receção de uma mensagem; Pedidos de resultados (*Result Request*) - permitem a solicitação de um resultado; Entrega de resultado (*Result Delivery*) - transportam os resultados para os seus destinatários.

Suponha-se que um cidadão vai a uma consulta médica com o médico pessoal. Devido a alguns sintomas, o médico solicita alguns exames médicos e a opinião de um especialista, reencaminhando o paciente para o Hospital.

O comportamento das Entidades que participam na prestação deste serviço em particular será semelhante ao que a seguir se descreve. O médico utiliza o SIL da clínica para realizar os passos necessários para prescrever o exame e reencaminhar o paciente para o Hospital (1). Uma vez que o SIL da clínica não tem o conhecimento necessário para realizar os pedidos, solicita-os à Entidade Clínica a que está associado (2). Até este ponto, nenhum dos elementos da arquitetura foi utilizado. Mais, o ator que solicitou o serviço não está ciente do recurso à Entidade por parte do SIL da Clínica, isto é totalmente transparente para o médico e para o paciente.

Depois de receber o pedido do seu SIL, a Entidade da Clínica analisa o pedido e determina que o pedido não pode ser decomposto (pelo menos por esta Entidade em particular). Neste ponto a Entidade pesquisa por Entidades que possam prestar o serviço (3) e recebe uma lista de possíveis prestadores (4). Depois de receber a lista, a Entidade Clínica escolhe a melhor opção, neste caso escolhe, para dar continuidade à prestação de serviço, o serviço prestado pelo Ministério da Saúde (MS) (5). Depois de receber o pedido, a Entidade do MS analisa-o, decompõe o serviço em serviços mais simples, encontra quem presta estes serviços e solicita-os. A Entidade começa por solicitar o exame médico à Entidade do Laboratório Clínico (LC) (6). A Entidade LC recebe o pedido, analisa-o e determina que o seu SIL pode prestar o serviço. Então, o pedido é reencaminhado para o SIL do LC (7) que realiza o serviço e devolve à Entidade LC os resultados (8). Após a conclusão do serviço, a Entidade LC envia uma mensagem de notificação à Entidade MS (9) contendo a localização do resultado. A Entidade MS retoma a execução do *workflow* e solicita uma consulta de especialidade ao Hospital (10).

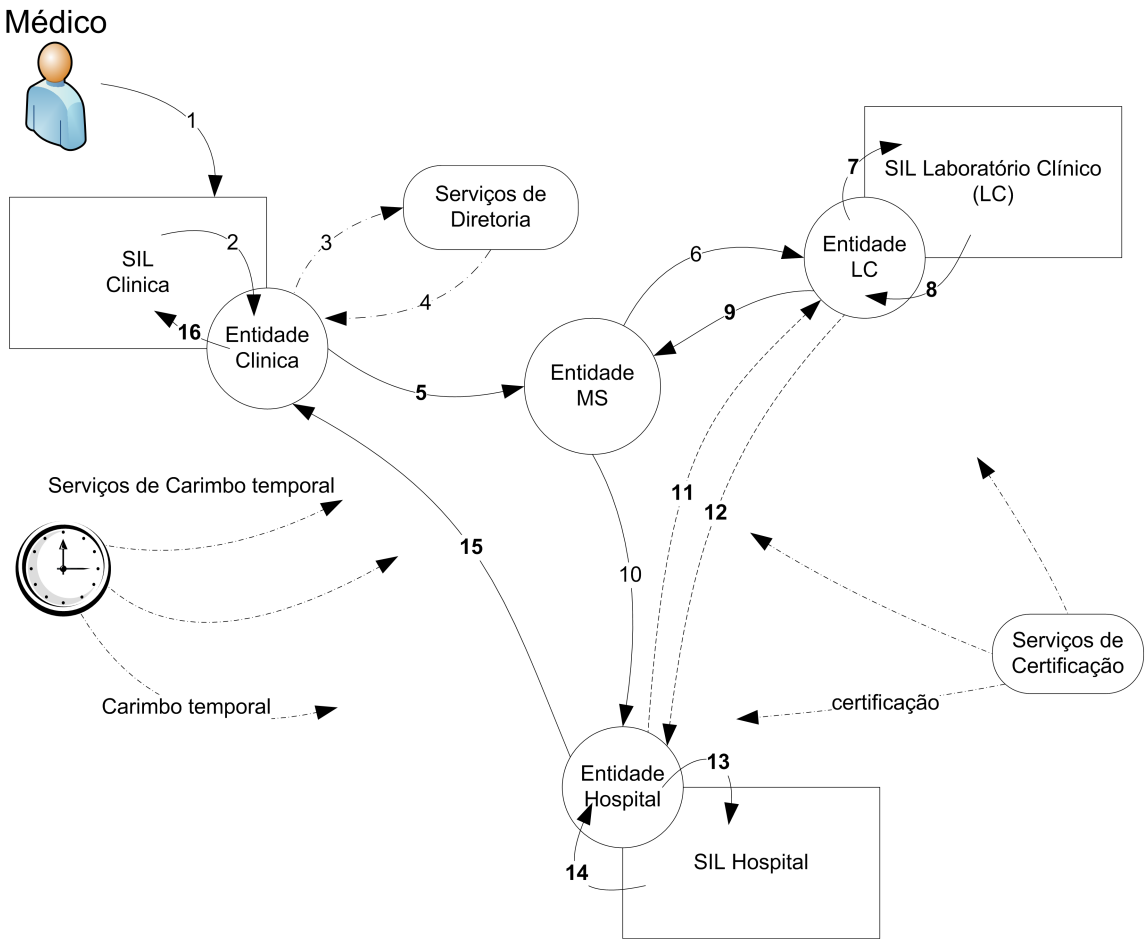

Figura 3.2: Arquitetura base e representação de prestação de serviços genérica.

Neste ponto, a Entidade Hospital determina a existência de exames e solicita-os à Entidade LC (11), os exames médicos são enviados para a Entidade Hospital (12). De notar que, nesta altura, a Entidade LC conhece o destino final dos exames clínicos e, consequentemente, pode cifrá-los de forma a que apenas a Entidade destinatária seja capaz de os ler, assegurando desta forma a confidencialidade e a privacidade dos resultados. Assim, e embora esteja envolvido num passo intermediário do *workflow*, a Entidade MS não tem acesso aos exames clínicos. Isto é uma característica-chave da arquitetura. A Entidade Hospital reencaminha o pedido e os exames para o SIL a que está associada (13). Depois do serviço ser realizado, o SIL do Hospital informa a sua Entidade (14), que, por sua vez, informa a Entidade Clínica que o serviço foi concluído (15). A Entidade Clínica conclui o serviço informando o SIL da Clínica (16).

Vamos identificar outra característica base da arquitetura que pode ser encontrada no exemplo: ambos os pedidos de serviço (exames clínicos e consulta médica no Hospital), para serem prestados e concluídos, necessitam da presença física do paciente, o que significa que existe uma diferença temporal entre o pedido inicial do serviço e a sua conclusão efetiva. Esta realidade não é problemática para a arquitetura, uma vez que apenas serve de intermediária

entre sistemas. Neste caso particular, os atores que interagem com o SIL do Hospital e com o SIL do Laboratório Clínico devem contactar o paciente para agendar os exames e a consulta médica, respetivamente (este processo deve ser gerido pelos SILs).

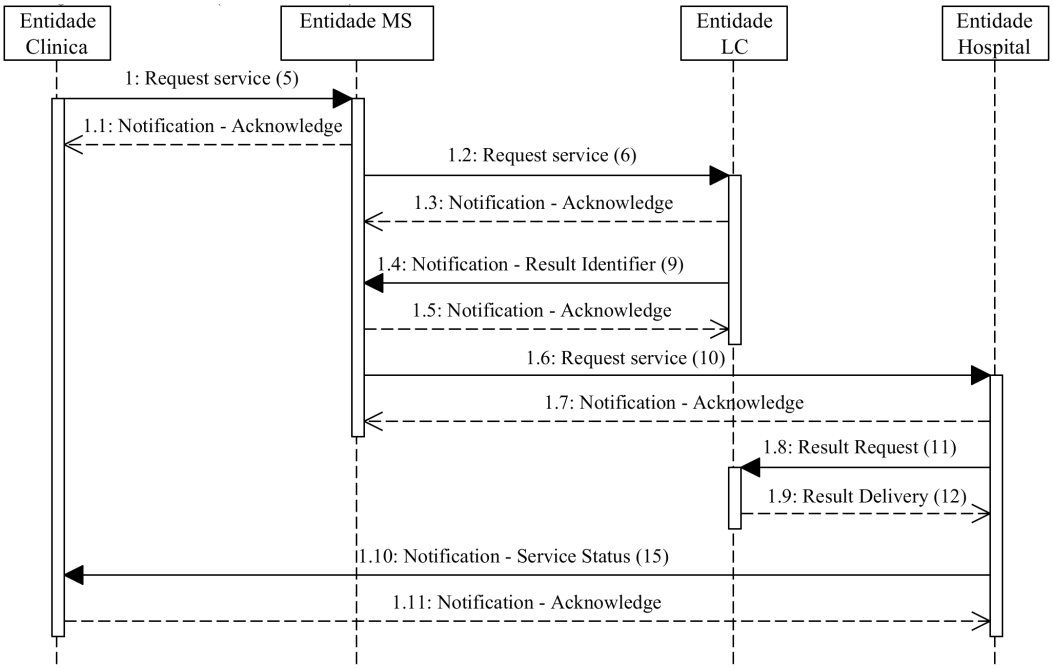

Figura 3.3: Diagrama UML de sequências que representa a troca de mensagens entre Entidades.

Antes de abordar as estruturas de dados que suportam a descrição do serviço e a acreditação/autorização/identificação, vamos analisar a troca de mensagens que ocorre entre as Entidades no exemplo. É possível seguir esta análise através do diagrama de sequências apresentado na Figura 3.3. Como mencionado na descrição, nem todas as mensagens foram incluídas na Figura 3.2. No passo 5 do exemplo, um *Request Service* é enviado da Entidade da Clínica para a Entidade MS, que responde com uma *Notification* do tipo *Acknowledge* reconhecendo a aceitação da prestação do serviço. O processo continua e, no passo 6, outra mensagem *Request Service* é utilizada, desta vez da Entidade MS para a Entidade LC. Novamente uma mensagem *Notification* do tipo *Acknowledge* é utilizada na resposta. Após a conclusão do serviço, a EntidadeLC envia uma mensagem *Notification* do tipo *Result Identifier*, passo 9, com a informação sobre a localização dos dados. A Entidade LC responde com a mensagem *Notification*, do tipo *Acknowledge*, acusando a receção do endereço do resultado. No passo 10, outra solicitação de serviço é enviada da Entidade MS para a Entidade Hospital. Uma vez que os resultados dos exames clínicos já são conhecidos, a mensagem enviada já incorpora a localização destes. Como resposta, é enviada uma mensagem *Notification* do tipo *Acknowledge*. No passo 11 a Entidade Hospital solicita os exames à Entidade LC através de uma mensagem do tipo *Result Request*, que responde utilizando um *Result Delivery*. Finalmente, a Entidade Hospital envia uma mensagem *Notification* do tipo *Service Status* no passo 15, que tem como resposta uma mensagem *Notification* do tipo *Acknowledge*. No diagrama de sequências da Figura 3.3 podemos observar todos os tipos de mensagens que podem ser

trocadas entre duas Entidades da arquitetura e de que forma são utilizados para a prestação de serviços compostos.

3.3.2 Modelo de segurança

Uma das principais preocupações no que concerne a integração de serviços e principalmente quando envolve a Administração Pública é a capacidade de proteger os dados dos intervenientes nas diferentes utilizações do sistema. A AP deve manter os dados dos seus constituintes protegidos. Além disso, os dados da AP devem por vezes ser mantidos em sigilo, mesmo daqueles a que se referem (e.g. investigações policiais). As entidades privadas têm também este tipo de preocupações, quer ao nível dos seus dados, quer ao nível dos dados dos seus clientes. Finalmente, o público em geral, que é particularmente desconfiado sobre a utilização que é feita dos seus dados (mantida quer por entidades públicas quer por entidades privadas) e relativamente àquela que está em trânsito. Isto significa que a segurança é fundamental em arquiteturas de integração.

Nesta secção identificamos as questões de segurança que resultam da arquitetura e apresentamos o modelo de segurança para os enfrentar.

Questões de segurança

A arquitetura de integração tem um conjunto de questões de segurança que deve ser analisado e resolvido, uma análise mais genérica publicada pelo Autor pode ser encontrada em [43].

Primeiro, uma entidade, como um recetor de pedidos, é abordada de uma de duas formas: através de um *Request Service* ou através de um *Result Request* (ver Figura 3.4). Ambos os tipos de pedidos podem ser realizados por qualquer entidade dentro da arquitetura. Uma vez que não existem restrições no que diz respeito ao acesso às entidades, estes pedidos podem também estar acessíveis a qualquer peça de software que consiga criar as mensagens para comunicar com a entidade. Como forma de proteção, a entidade deve verificar a autorização do solicitador do serviço.

Segundo, o caminho do *workflow* pode forçar algumas limitações à prestação do serviço, por exemplo: conflitos de interesse entre duas entidades que estão envolvidas no mesmo processo de prestação do serviço (e.g. uma consultora a avaliar propostas para um projeto para o qual se candidatou). As entidades devem estar preparadas para atuar (verificar a autorização de participação no *workflow*) de acordo com estes casos e responder de forma apropriada.

Terceiro, existem algumas preocupações que devem ser abordadas pelo solicitador do serviço ou resultado. Para pedir um serviço, é necessário verificar se a entidade a quem o mesmo será solicitado está acreditada para o prestar (o que significa que uma terceira entidade de confiança confirma que uma entidade não só é capaz de prestar um serviço mas também tem a qualificação necessária para o fazer). Neste caso (solicitar um serviço), a entidade solicitadora deve ser capaz de identificar qualquer restrição ao nível do *workflow* que possa impedir a participação de outra entidade no *workflow*.

Finalmente, devem ser consideradas algumas precauções aquando da solicitação de um resultado. Uma vez que uma entidade não sabe para quem está a produzir um resultado no momento em que o produz, então é incapaz de explorar as transformações de cifragem/decifragem de forma a assegurar a confidencialidade e a privacidade do mesmo quando em trânsito para outras entidades. Uma vez que este é o caso por omissão na arquitetura, então a entidade

mantém o resultado produzido até este ser explicitamente requisitado pela entidade que necessita dele. Esta necessidade de solicitar o resultado adiciona algumas preocupações à entidade que dele necessita: primeiro, verificar a validade do resultado (ou seja, verificar se o autor do resultado tem qualificação para produzir aquele resultado. Por outras palavras, verificar se a entidade está acreditada para prestar o serviço que deu origem ao resultado.); segundo, se a entidade que requer o resultado tem de produzir um novo resultado com base neste resultado anterior, então tem de identificar a entidade que originalmente providencia o resultado (é importante para responsabilização).

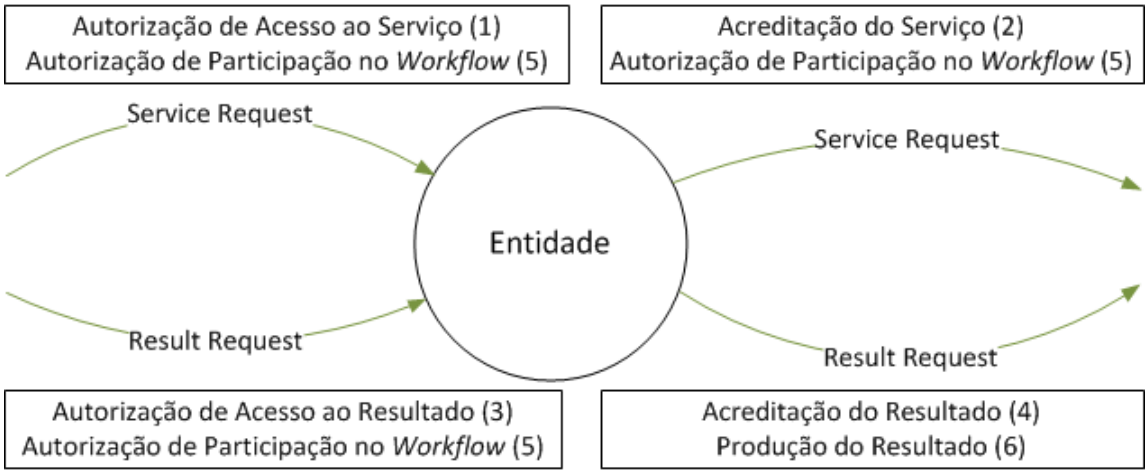

Figura 3.4: Tipos de mensagens e preocupações de segurança relacionadas.

Resumindo, deve-se assegurar que (ver Figura 3.4):

1. Uma entidade que solicita um serviço está autorizada para o fazer.
2. Uma entidade que disponibiliza um serviço está acreditada para o fazer.
3. Uma entidade que requer um resultado está autorizada para o obter.
4. Uma entidade que produz um resultado está acreditada para o oferecer.
5. Uma entidade que participa no *workflow* está autorizada para participar nesse mesmo *workflow*.
6. Uma entidade que disponibiliza um resultado é a mesma entidade que o produziu.

Mecânica do modelo

Para abordar as questões de segurança identificadas na secção 3.2 utilizamos a estrutura *Certificate* descrita na secção 3.3.3. Para a Autorização de Acesso ao Serviço e para a Autorização de Acesso ao Resultado (1 e 3) a entidade prestadora compara a classificação (c_r) contida no certificado de autorização da entidade requerente com a classificação (c_p) do certificado de acreditação que a entidade possui para o serviço que foi pedido, com a obtenção dos seguintes resultados: Não autorizado, se $c_r < c_p$; Autorizado, se $c_r \geq c_p$.

Para abordar a Autorização ao nível do *Workflow* (5), a entidade utiliza a informação contida no *ServiceRequest* (ver Figura 3.9) sobre as entidades que já contribuíram para o *workflow* e os seus certificados. Esta avaliação baseia-se num conjunto de políticas de autorização da entidade que presta o serviço e nos certificados de autorização das entidades que participaram no *workflow*.

Os certificados, de facto as instâncias desta estrutura com o propósito de acreditação, são utilizados para abordar tanto a Acreditação do Serviço como a Acreditação do Resultado (2 e 4). Para a Acreditação do Serviço, isto é conseguido quando a entidade requerente está à procura de uma entidade a quem possa delegar o serviço (ou parte dele). Isto é realizado através da verificação das credenciais e da confiança nas entidades certificadoras que as emitiram. O processo para a Acreditação do Resultado é idêntico – a verificação incide nas credenciais e na confiança nas entidades certificadoras que as emitiram. A única diferença é o facto de neste caso a entidade que retém o resultado já ser conhecido e o serviço já ter sido executado.

Para verificar se o resultado é produzido pela entidade que o está a facultar – Produção do Resultado (6) – é necessário verificar se o endereço está digitalmente assinado pela mesma entidade.

Para além abordar as questões levantadas na secção 3.3.2 o modelo deve abordar as propriedades de segurança identificadas na secção 3.1.5. Todos os intervenientes no processo devem ser identificados, incluindo os cidadãos e entidades. O requerente do serviço original (um cidadão) autentica-se no SIL. O SIL deve transmitir esta informação para a entidade associada, que a deve armazenar na forma de um resultado. Por sua vez, as entidades identificam-se quando dão início à comunicação entre si.

A confidencialidade da informação que flui na arquitetura é conseguida de duas formas: cifrando a comunicação entre entidades; e, através da cifra da informação contida nas mensagens. O primeiro método garante que mesmo que os pacotes em circulação sejam capturados seja bastante difícil perceber o seu conteúdo. Uma vez que teremos sempre uma arquitetura do tipo cliente/servidor (quando nos centramos em apenas uma comunicação) a comunicação segura é obtida pelo uso das chaves assimétricas das entidades que estão envolvidas no processo. O segundo método também utiliza credenciais. Neste caso, a chave pública da entidade que recebe a mensagem, que é disponibilizada na sua credencial. Desta forma garantimos que a única entidade que tem acesso à informação na mensagem será aquela a quem a credencial pertence.

A arquitetura utiliza assinaturas digitais para abordar a propriedade integridade. Todo o conteúdo das mensagens, após a sua chegada, é verificado para controlo da integridade. No caso de a integridade ter sido violada, então a mensagem em particular é descartada e a entidade requerente é notificada. Este método é também utilizado para averiguar que nenhuma ação é executada sem ator. Por outras palavras, se uma mensagem está assinada digitalmente, então, a não ser que a chave privada esteja comprometida, apenas um ator a pôde assinar. Isto torna impossível que alguém (ou alguma coisa) que tenha contribuído para uma ação em particular possa negar o seu envolvimento no processo, abordando a propriedade de não-repúdio. Mais, uma vez que todas as mensagens trocadas entre entidades estão digitalmente assinadas, é muito simples seguir uma ação até ao seu autor, atingindo a propriedade da rastreabilidade.

3.3.3 Componentes

Esta secção destina-se a apresentar os componentes da arquitetura que propomos. Começaremos por apresentar os serviços de suporte à arquitetura e as estruturas de dados que lhes estão associadas (secção 3.3.3) e que permitem garantir a prestação dos serviços necessários ao bom funcionamento da arquitetura. Em seguida, apresentamos as mensagens e a ordem pela qual estas mensagens são utilizadas na comunicação entre entidades (secção 3.3.3) de forma a permitir a sua colaboração durante a prestação dos serviços. Por fim, concluiremos com a descrição de uma proposta de organização interna das entidades (secção 3.3.3).

Serviços e estruturas de dados

De forma a permitir a prestação segura de serviços, publicar e localizar serviços disponíveis e validar a sequência do *workflow* em tempo real, três serviços de suporte à arquitetura devem estar disponíveis: Serviço de Carimbos temporais; Serviço de Diretoria; e Serviço de Certificação. Estas infraestruturas são apresentadas nas próximas secções.

Serviço de Carimbos temporais. Todas as mensagens trocadas entre as Entidades dentro da arquitetura têm um carimbo temporal associado. Isto permite uma colocação temporal da geração da mensagem e consequente rastreabilidade das trocas de mensagens. O serviço de carimbo temporal adiciona o carimbo temporal à mensagem e fica com um registo da data e hora de quando este serviço foi solicitado.

Serviço de Diretoria. O Serviço de Diretoria tem como principal objetivo disponibilizar a descrição dos serviços que podem ser utilizados pelas Entidades. Para tal, são disponibilizados repositórios de serviços onde as entidades publicam (ou removem) as descrições dos serviços e onde podem pesquisar e localizar os serviços de que necessitam.

A estrutura de dados que suporta esta descrição denomina-se *Service Description* (ver Figura 3.5) cujo principal elemento é a classe *Service*. São as instâncias desta estrutura que estão disponíveis nos repositórios de serviços que constituem o Serviço de Diretoria.

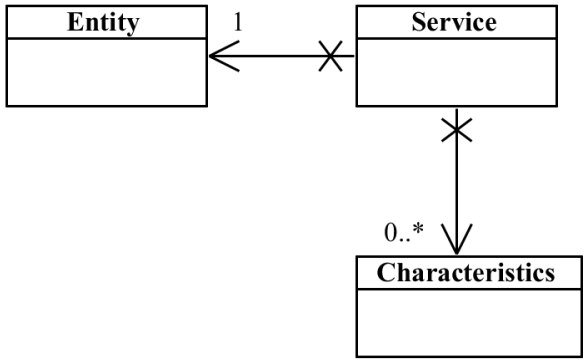

Figura 3.5: Diagrama UML de classes da descrição do serviço

A estrutura é composta por três classes: *Service*; *Entity*; e, *Characteristic*. A classe *Service* representa o serviço e deve conter a descrição do serviço publicado, os dados necessários para a prestação do serviço e o resultado esperado. A classe *Entity* identifica a entidade que

publicou o serviço e deve incluir as credenciais que demonstram que está acreditada para poder executar o serviço. Como veremos mais adiante, as credenciais são concretizadas por certificados emitidos por Serviços de Certificação.

A classe *Characteristic* é utilizada para armazenar um conjunto de características do serviço que permitem definir a ordenação de prestadores de serviços pelos solicitadores. Isto permite a ordenação dos prestadores do serviço baseando-se em características como o custo, tempo de prestação previsto, intervenção humana, qualidade do serviço, etc. Este conjunto de características deve estar definido por toda a arquitetura, permitindo a comparação entre serviços e a respetiva ordenação de acordo com o critério ou critérios preferidos.

Serviço de Certificação. O principal objetivo do Serviço de Certificação é suportar a identificação, a acreditação e a autorização de entidades. Este serviço permite igualmente concretizar um conjunto de propriedades de segurança: confidencialidade, integridade e não-repúdio.

O serviço é responsável pela emissão e revogação de certificados de chave pública para cada serviço prestado por uma entidade (certificados de acreditação) e de certificados de chave pública para acesso a serviços (certificado de autorização). A emissão de um certificado é fundamental para uma Entidade para que se torne num membro válido da arquitetura, uma vez que apenas Entidades com certificados de acreditação são capazes de fornecer serviços e Entidades com certificados de autorização são capazes de consumir serviços.

As estruturas de dados das Figuras 3.6 e 3.7) suportam a acreditação do par {Entidade, Serviço} e a autorização para acesso a serviços, respetivamente. Esta estrutura de dados foi definida de forma a que o serviço de certificação suportasse as três propriedades mencionadas inicialmente:

identificação: ambas as estruturas permitem identificar uma entidade para o registo e autorização baseados na identidade;

acreditação: para certificar que a entidade é capaz e está autorizada para fornecer o serviço que publicou; e,

autorização baseada em funções: para fornecer atributos para permitir que as entidades tomem decisões de autorização baseadas em funções.

Para atingir todos os objetivos, a estrutura *Certificate* foi definida como podemos observar nas Figuras 3.6 e 3.7. As duas estruturas têm um conjunto de classes em comum: *Certificate*; *Certifier*; *Subject*; *Validity*; *Service*; *Type*; e, *Classification*.

As classes *Certifier* e *Subject* representam a identidade e os elementos de prova da mesma do emissor do certificado e da entidade para quem o certificado é emitido, respetivamente. Ambas as classes devem conter as chaves públicas correspondentes. A classe *Validity* define o período em que o certificado é válido. Este conjunto de classes é suficiente para abordar o objetivo de identificação de uma Entidade.

Para abordar a acreditação de um serviço de uma Entidade as classes *Service*, *Classification* e *Type* têm de estar instanciadas (ver Figura 3.6). A classe *Type* identifica o tipo de certificado (autorização ou acreditação), enquanto a classe *Service* descreve o serviço que a Entidade está certificada para fornecer. A classe *Classification* determina o menor nível de acesso para que uma Entidade posso solicitar o serviço.

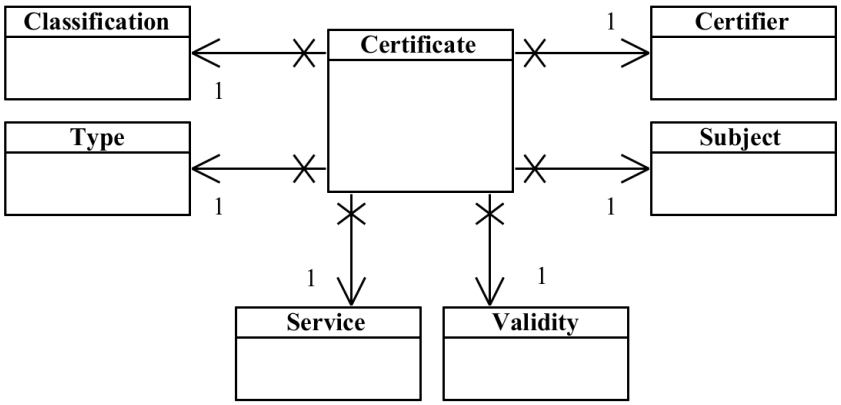

Figura 3.6: Diagrama UML de classes do certificado de acreditação

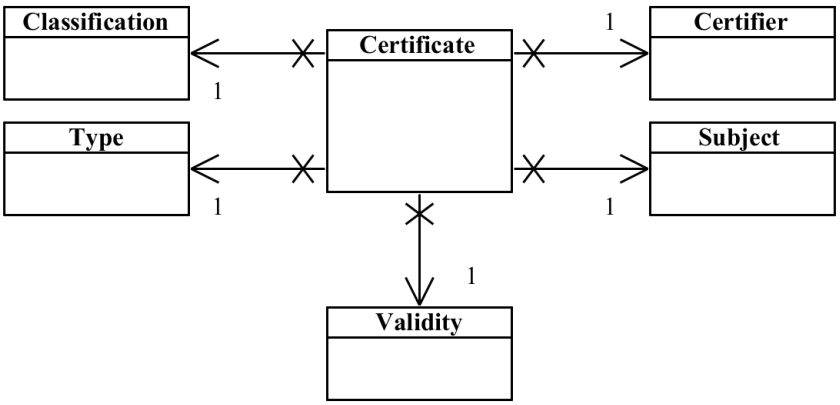

Figura 3.7: Diagrama UML de classes do certificado de autorização

Para abordar a autorização o certificado deve ter apenas as classes *Classification* e *Type* instanciadas (ver Figura 3.7). Neste caso a classe *Type* identifica um certificado de autorização. A instância da classe *Classification* representa o nível de acesso da Entidade. Na prática este nível de acesso representa a habilitação de segurança da entidade quando requer o serviço. A classificação deve ser acordada entre o prestador do serviço e a entidade certificadora ou entidades certificadoras que emitem os certificados. Apenas desta forma é possível garantir que as entidades certificadoras emitem certificados para autorização com a habilitação de segurança correta para as entidades que o solicitem.

Mensagens

As mensagens são uma das partes mais importantes da arquitetura. Elas permitem a troca de informação fornecendo uma linguagem comum através da qual as entidades comunicam. Definimos 4 tipos de mensagens que suportam todas as interações necessárias no âmbito de arquitetura: *Request Service*, *Result Request*, *Notification*, e *Result Delivery*. Embora cada tipo de mensagem tenha sido definido com uma finalidade especifica, que abordaremos mais tarde, todas partilham a mesma estrutura base, como demonstra a Figura 3.8. Todos os tipos

de mensagem têm um carimbo temporal e são assinadas digitalmente pela entidade que as envia.

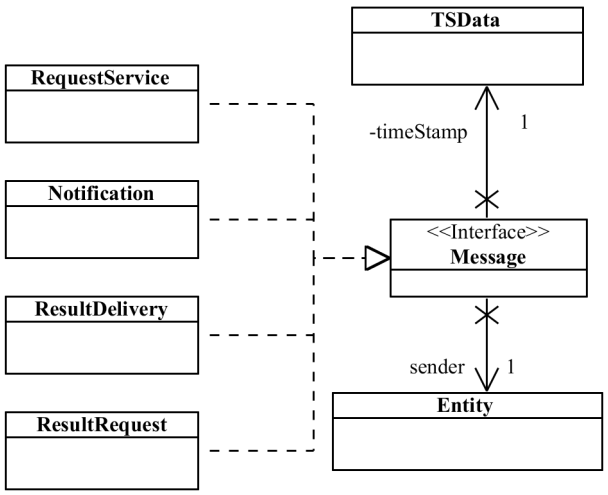

Figura 3.8: Diagrama UML de classes da mensagem.

Request Service A mensagem *Request Service* é o tipo de mensagem que inicia a interação com uma entidade. Tem a sua origem na Entidade requerente, que pode ou não ter um SIL associado, e é endereçado a uma Entidade prestadora de um serviço. Na Figura 3.9 é apresentada a representação da estrutura deste tipo de mensagem. O *Request Service* contém um conjunto de atributos que caraterizam um pedido e identificam o serviço a ser solicitado e o destinatário ou destinatários finais do resultado, caso existam.

A mensagem tem como elemento principal a classe *RequestService* que contém um identificador que representa o número do processo da entidade requerente ou do SIL associado. Deve conter igualmente informação detalhada sobre o serviço a ser solicitado sendo suportado pelo tipo de dados *Service* descrito em 3.3.3.

Há ocasiões em que o requerente não é o destinatário do resultado. Por exemplo, quando um aluno requer a declaração de rendimentos para entregar durante o processo de candidatura a uma bolsa de estudos. Neste caso os Serviços Sociais e não o requerente (o aluno) são os destinatários finais do resultado do pedido. Para suportar estas situações, a estrutura permite a identificação de destinatários múltiplos. Cada destinatário é identificado pela instanciação da classe *Addressee*. Para além disso, para cada destinatário, uma lista de contactos pode ser providenciada: e-mail, caixa postal, residência, Serviço Web, etc.

Cada *Request Service* pode ser também composto por um conjunto de elementos com a mesma estrutura (*RequestService*), que armazenam os dados sobre os serviços anteriormente solicitados no âmbito do serviço complexo que está a ser prestado. Isto permite que cada entidade esteja ciente do histórico do *workflow*, o que é útil para validar a sua participação, como iremos ver em detalhe na secção 3.3.2.

Finalmente, alguns serviços requerem dados adicionais. Por exemplo quando um médico pede uma segunda opinião a um colega, os exames clínicos realizados pelo paciente não são repetidos – são adicionados ao pedido de serviço e enviados juntamente com o pedido. Neste

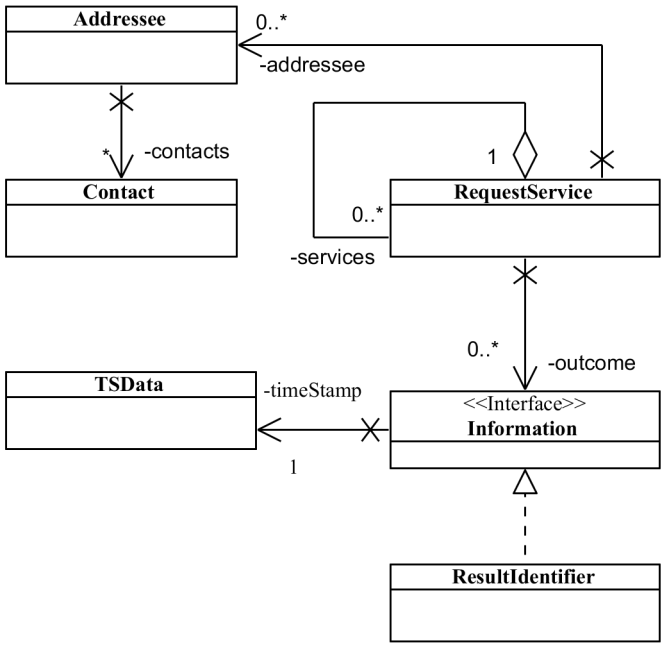

Figura 3.9: Diagrama UML de classes da mensagem *Request Service*.

caso os dados são conhecidos ou já foram produzidos aquando o pedido. Esta informação é adicionada ao *ServiceRequest* na forma de uma *Information* realizada como um *ResultIdentifier*. Este tipo de dados não contém os dados em si, apenas contém a informação em como retirar o resultado, ou seja, a localização e a descrição da informação que está na posse de quem a produziu. Esta estrutura de dados é também assinada digitalmente pelo prestador que a criou, o que assegura a sua integridade e autoria. Além disso, uma vez que tem um carimbo temporal, a hora da criação do *ResultIdentifier* também está assegurada, permitindo validar a data e a hora da sua criação, sendo útil em diversas situações - por exemplo: assegurar que um resultado ainda se encontra válido, construir o histórico da prestação do serviço, etc.

Notification A mensagem do tipo *Notification* (ver Figura 3.10) foi definida com o objetivo de fornecer um mecanismo para que as entidades procedessem à alteração do estado dos pedidos realizados e que comporta três funcionalidades básicas: reconhecer a receção de um pedido de serviço; informar uma entidade requerente da alteração de estado na prestação do serviço; e, informar uma entidade sobre a localização de um resultado. Este tipo de mensagem tem como elemento principal o *Notification* que identifica o pedido efetuado e deve incluir a assinatura digital do prestador do serviço. Para permitir a validação da assinatura, a identificação e os certificados do prestador devem estar incluídos na mensagem.

As funcionalidades são abordadas através de uma interface que garante a consistência da mensagem. É a implementação que diferencia a mensagem em relação ao objetivo. Apenas uma das implementações pode estar presente num momento no tempo. A classe *Acknowledge* contém informação sobre a identificação local do processo; a classe *ServiceStatus* aborda todas as alterações de estado que um serviço pode sofrer enquanto está a ser processado por uma entidade, incluindo eventuais erros; e a classe *ResultIdentifier* (já apresentada) que representa

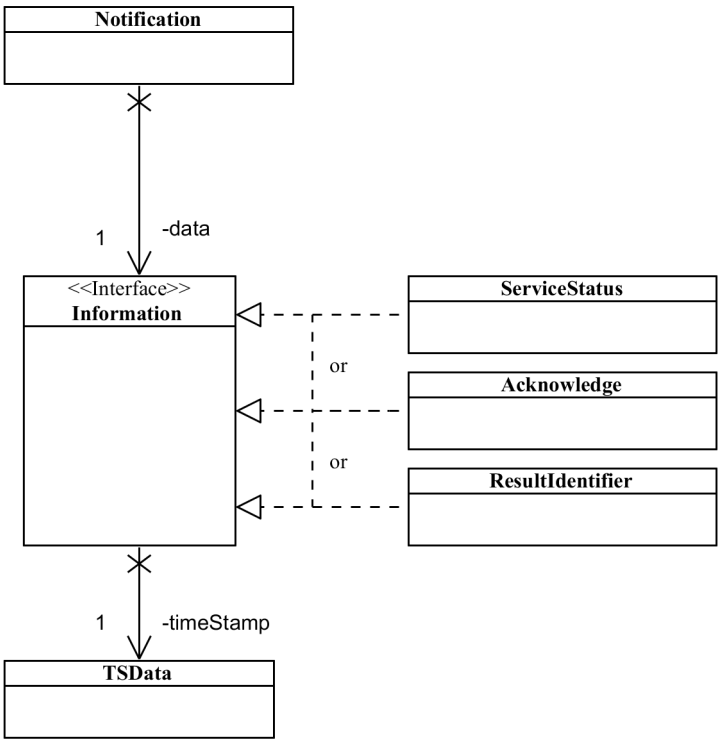

Figura 3.10: Diagrama UML de classes da mensagem *Notification*.

os dados que podem ser acedidos, como lhes aceder, e quem foi o produtor do resultado.

Result Request Na Figura 3.11 temos a representação do tipo de mensagem *Result Request*. O único objetivo deste tipo de mensagem é solicitar o envio de um resultado já produzido, que tinha sido obtido por um pedido de serviço anterior. Por esta razão, este tipo de mensagem é enviado para a entidade que produziu o resultado e que o armazena.

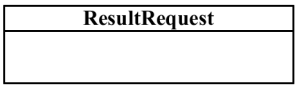

Figura 3.11: Diagrama UML de classes da mensagem *Result Request*.

Uma mensagem *Result Request* contém o elemento *ResultDetail*. Este elemento faz referência ao processo da entidade (entidade recetora da mensagem) que mantém o resultado, ao número do processo na entidade requerente local, este número diz respeito ao processo na entidade que está a solicitar o resultado. O elemento *ResultRequest* contém igualmente a localização do recurso, e a assinatura digital da entidade requerente. Para permitir a verificação da validade da assinatura, os certificados e a identificação estão contidos na mensagem. Finalmente, a mensagem do tipo *Result Request* tem um carimbo temporal que permite a rastreabilidade da prestação do serviço, como podemos observar na Figura 3.8.

Result Delivery A Figura 3.12 representa o tipo de mensagem que é criada como resposta a uma mensagem do tipo *Result Request*. Contém a informação sobre o número original do processo, este contém o valor correspondente ao processo na entidade que solicitou o resultado. Um carimbo temporal está igualmente incluído. Finalmente, a estrutura inclui o resultado produzido, que é cifrado utilizando a chave pública da entidade que o solicitou. Isto permite uma correta entrega de dados ao requerente e garante a confidencialidade e a integridade do resultado.

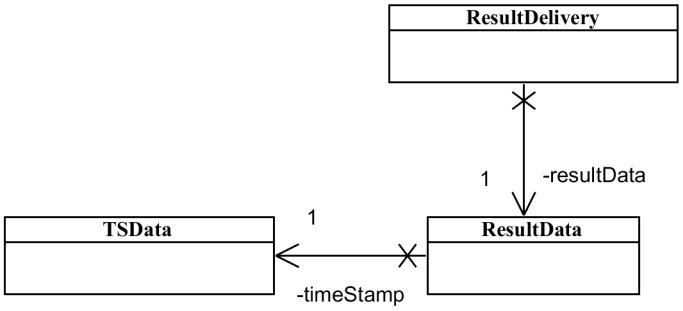

Figura 3.12: Diagrama UML de classes da mensagem *Result Delivery*.

Comunicação Toda a comunicação dentro da arquitetura é feita entre Entidades. Um SIL apenas pode aceder à arquitetura através de uma Entidade.

Nas secções anteriores mostrámos que existem 4 tipos de mensagens. A ordem pela qual aparecem no *workflow* é importante e está diretamente relacionada com o propósito com que foram definidas. Na Figura 3.13 pode ser observada a sequência-padrão de mensagens trocadas. Esta sequência é válida dentro do ambiente de *workflow* e com uma correta prestação de serviço.

Os serviços fornecidos pelas entidades são assíncronos. No entanto, as mensagens trocadas são síncronas. De forma a poder dar início ao processo de requisitar um serviço, a mensagem do tipo *Service Request* é gerada e enviada para a entidade que publicou o serviço. Em resposta, a entidade requerente recebe uma mensagem do tipo *Notification*. Quando uma entidade delega a prestação do serviço a outra entidade o delegante vai sendo atualizado através da receção de mensagens do tipo *Notification* que são enviados pela entidade em que delegou. A primeira mensagem deste tipo é uma mensagem *Acknowledge* que reconhece a aceitação do pedido efetuado, indicando a entidade a quem foi solicitado o pedido se compromete a realizar o serviço.

Finalmente, os restantes dois tipos de mensagens permitem transferir um resultado previamente produzido. Neste caso, um *Result Request* é criado pela entidade requerente baseando-se na localização que foi obtida anteriormente numa mensagem do tipo *Notification* e, como resposta, uma mensagem do tipo *Result Delivery* é enviada da entidade que produziu o resultado para aquela que o solicitou.

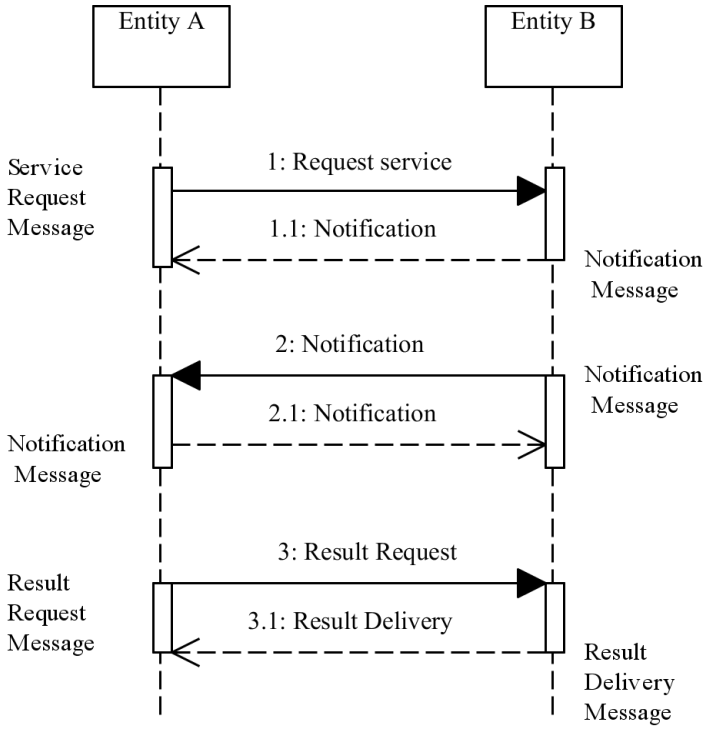

Figura 3.13: Diagrama UML de sequência que exemplifica o padrão de mensagens.

Entidades

A entidade é um elemento chave da arquitetura. A entidade é um representante de um organismo público ou privado e que é caraterizado por ser autónomo (é capaz de tomar decisões sem que existam intervenções externas); é reativa (responde a eventos ou estímulos de forma apropriada); é proativa (está ciente das alterações que ocorrem e adapta-se em conformidade); tem capacidade de interação profissional (comunica com os seus pares e com o SIL) para colaborar ou competir com outros. Uma entidade pode fornecer vários serviços simples ou complexos.

Comportamento Nesta subsecção vamos apresentar um possível comportamento para uma entidade (ver Figura 3.14). As Entidades que estão associadas a um SIL recorrem à atividade *Wraps Message* para transformar mensagens normalizadas em mensagens locais (mensagens que têm a sua origem no SIL).

Quando uma mensagem é recebida, a atividade *Receive Message* examina a mensagem, determina o seu tipo e redireciona-a em conformidade. Qualquer um dos três caminhos principais pode ser seguido, dependendo apenas do tipo da mensagem recebida. Quando um *Service Request* é recebido, a atividade *Assess Service Authorization* é chamada para verificar se a entidade que solicitar o serviço está autorizada a fazê-lo. Este teste é baseado nas credenciais da entidade requerente do serviço. A decisão relativa à participação no *workflow* é também formada neste passo, o que pode implicar numa análise de todas as entidades que previamente contribuiram para o *workflow*. Se o pedido de serviço passou estes controlos, o

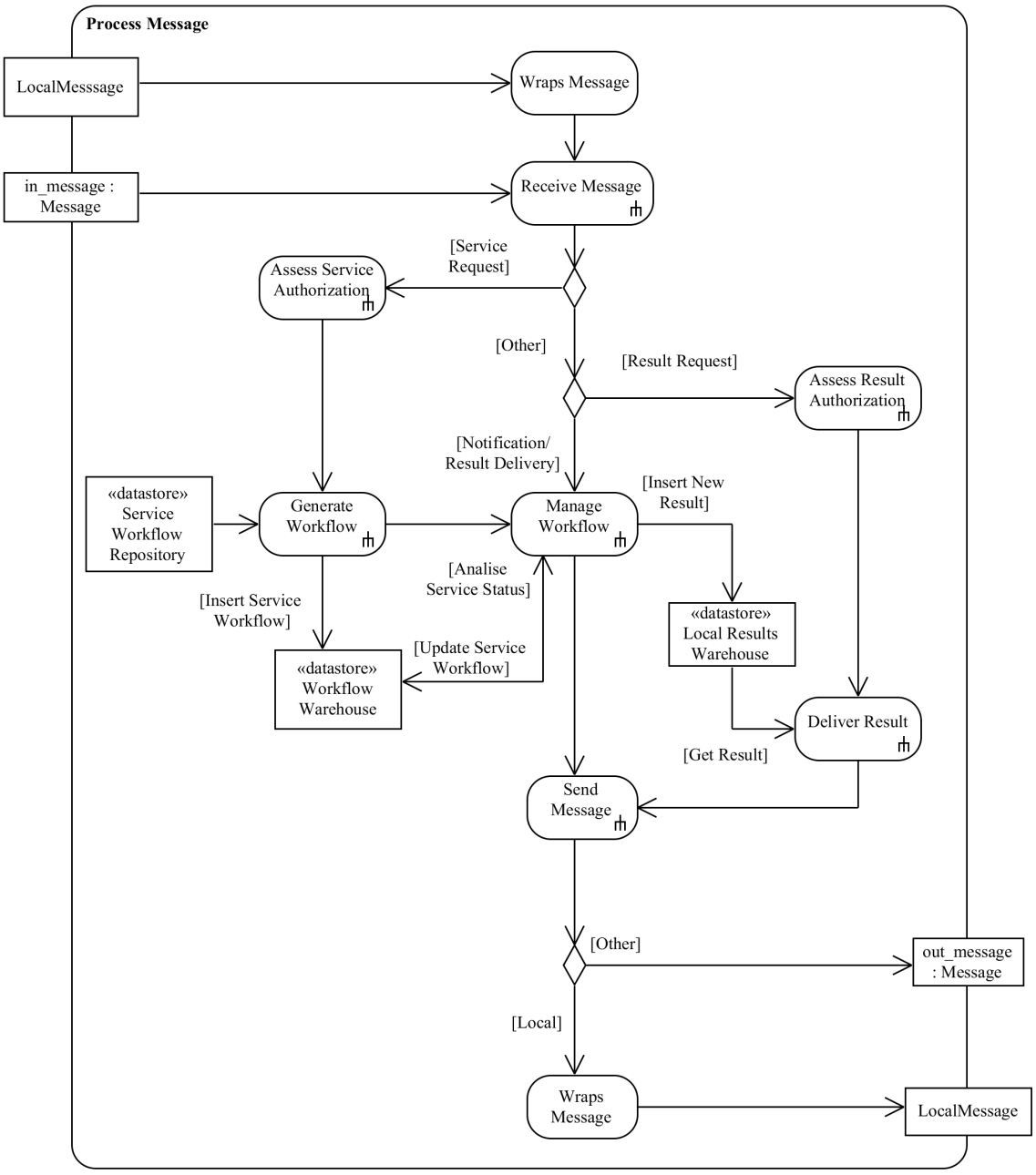

Figura 3.14: Diagrama UML de atividades do workflow interno de uma Entidade.

processo segue para a atividade *Generate Workflow*. A principal função desta atividade é ler o *Service Workflow Repository* (que armazena todos os *workflows* que esta entidade conhece), gera os passos do *workflow*, armazena-os no *Workflow Warehouse* e delega a execução para a atividade *Manage Workflow*. Embora assumamos a existência de um repositório de *workflows* de serviços, isto não significa que todos os *workflows* sejam armazenados estaticamente. De facto, a arquitetura não impede que a atividade componha dinamicamente o *workflow* usando qualquer informação a que tenha acesso. As atividades *Assess Service Authorization* e *Assess*

Result Authorization serão abordadas em maior detalhe em seguida.

Quando é recebida uma mensagem do tipo *Result Request*, a atividade *Receive Message* reencaminha a mensagem para a atividade *Assess Result Authorization*, que vai determinar se a entidade requerente está autorizada a aceder ao resultado e para fazer parte do *workflow*. Esta verificação baseia-se nas credenciais apresentadas pela entidade requerente. Se a entidade requerente for autorizada, então a atividade *Deliver Result* é invocada. Esta atividade obtém o resultado do *Local Results Warehouse* (repositório que contém todos os resultados que são produzidos pelo SIL da entidade) e reencaminha o resultado para a atividade *Send Message*. Esta atividade é responsável por todas as comunicações de saída.

Finalmente, os dois tipos de mensagem restantes (*Notification*, *Result Delivery*) são reencaminhados, pela atividade *Receive Message*, para a atividade *Manage Workflow* que é responsável por determinar o próximo passo, por inserir novos resultados (quando necessário), e por determinar qual a melhor entidade a quem solicitar um serviço. Para além de identificar o melhor candidato do grupo de possíveis prestadores de serviços, a atividade deve verificar se o prestador do serviço é capaz de realizar o serviço (está acreditada para esse efeito) e se está, igualmente, autorizado a fazer parte do *workflow*. Quando realiza um *Request Result*, deve verificar se a entidade que produziu o resultado está acreditada para gerar o resultado e, para além disto, se é a produtora do resultado. Isto permite identificar entidades que não estão a funcionar de forma adequada e/ou que estão a tentar aceder informação à qual não deveriam ter acesso. Para auxiliar nas suas funções, a atividade *Manage Workflow* tem acesso a todos os *workflows* que foram previamente gerados pela sua entidade e que estão armazenados no *Workflow Warehouse*. Tem também a capacidade de comunicar com o repositório de serviços (ver Figura 3.2) que, por simplicidade, não é apresentado em Figura 3.14.

Nas secções que se seguem apresentamos as atividades *Assess Service Authorization* e *Assess Result Authorization* e que permitem a identificação dos pedidos que serão aceites pela entidade. De notar que a entidade deverá descartar mensagens que não passem na avaliação efetuada por qualquer das atividades.

Autorização de acesso ao serviço O processo apresentado na Figura 3.15 tem como objetivo a avaliação da autorização da entidade que solicitou o serviço. Esta verificação é realizada na atividade *Assess Authorization* utilizando as credenciais do solicitador do serviço.

Outra das validações que é realizada nesta atividade está relacionada com a participação da entidade no *workflow* executado até ao momento. Para tal é necessário identificar todos os intervenientes na prestação do serviço até então. Isto é realizado na componente *Rebuild Backward Workflow Path* recorrendo à informação contida na mensagem que solicitou o serviço. Apenas se estas duas condições se verificarem positivas é que a autorização para o processamento do pedido é atribuída.

Autorização de acesso ao resultado O processo representado na Figura 3.16 é similar ao exposto na secção anterior. No entanto, o objetivo é distinto. Enquanto na atividade *Assess Service Authorization* o objetivo é verificar se um interlocutor poderá ter acesso a um serviço, nesta atividade o objetivo é permitir ou negar o acesso ao resultado previamente produzido.

Este processo inicia-se com a verificação da autorização de acesso ao resultado recorrendo às credenciais da entidade solicitadora. Em seguida, e como o resultado apenas terá como destinatário o seu solicitador, a autorização de participação do interlocutor no *workflow* é avaliada.

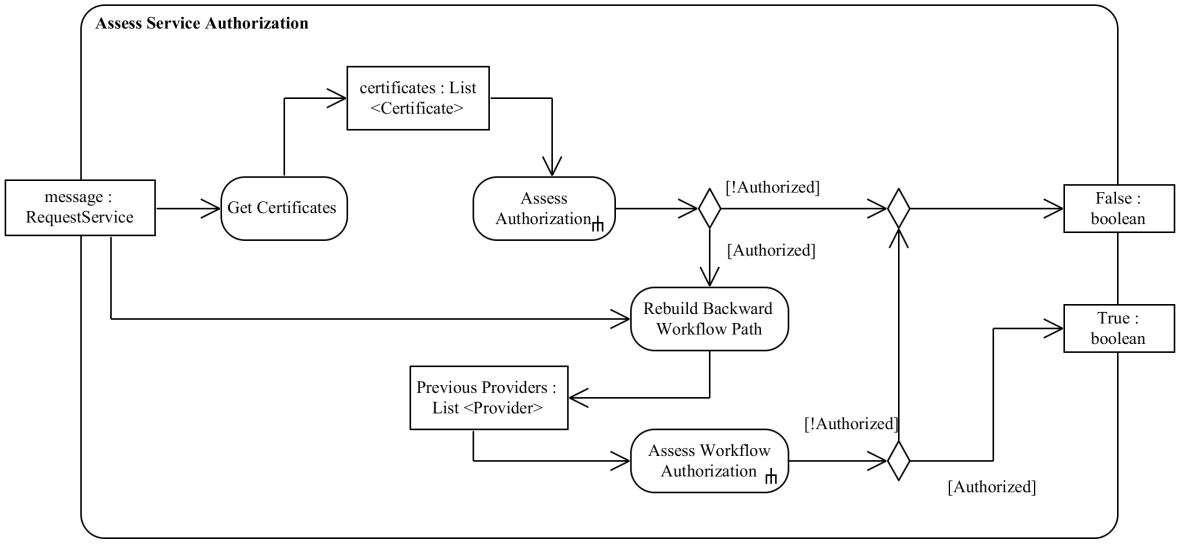

Figura 3.15: Diagrama UML de atividades para o processo de verificação da autorização de acesso a um serviço.

Tal como na componente *Assess Service Authorization*, apenas se o resultado de ambas for positivo é que a solicitação do resultado pela entidade será autorizada, sendo o resultado enviado para o agente que o solicitou.

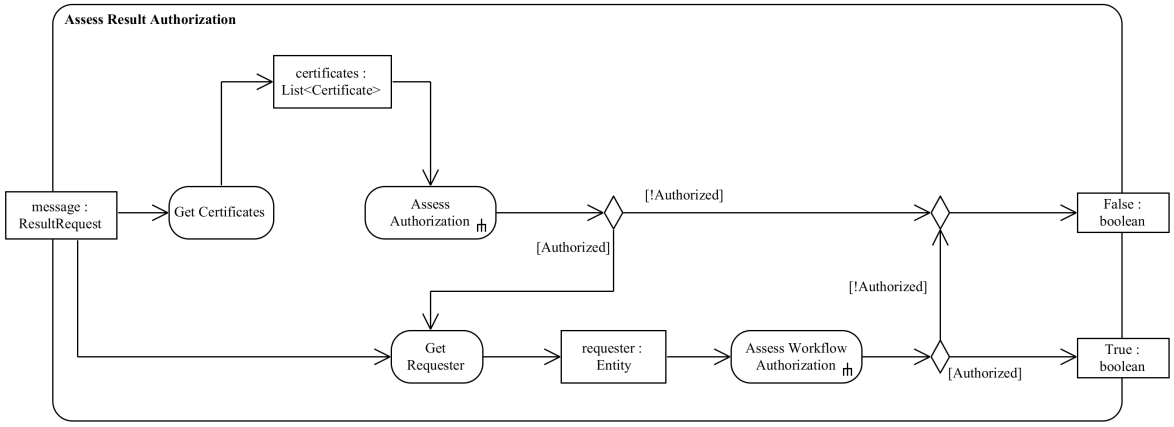

Figura 3.16: Diagrama UML de atividades para o processo de verificação da autorização de acesso a um resultado.

3.3.4 Observância dos objetivos

Previamente, estabelecemos um conjunto de objetivos que, acreditamos, são fundamentais para a nossa arquitetura de integração: integração de serviços, concorrência, versatilidade, escalabilidade e segurança. Nesta secção vamos demonstrar que a arquitetura verifica estes objetivos.

A arquitetura é por desenho versátil, uma vez que permite a adição/remoção de serviços através da publicação/remoção destes mesmos serviços nos repositórios de serviços. Para além disso, as entidades são adicionadas à arquitetura através da publicação de pelo menos um serviço acreditado ou são removidas como consequência da remoção de todos os seus serviços acreditados dos repositórios de serviços.

Cada ramo da AP ou entidade privada que deseje publicar um serviço na arquitetura pode fazê-lo. A utilização do serviço está dependente da certificação do par {serviço, entidade}. As entidades tomam as suas decisões independentemente das entidades que prestam o serviço, uma vez que apenas estão dependentes do *workflow* que deve ser seguido. A escolha das entidades que devem dar seguimento à prestação do serviço está apenas dependente dos critérios definidos pela entidade ou cliente requerente. Isto permite a criação de diferentes caminhos durante a prestação de dois serviços similares, o que permite a prestação concorrente de serviços.

Como vimos na secção 3.3.3, a integração de serviços é conseguida através da composição e decomposição de serviços efetuada pelas entidades, recorrendo a serviços que são prestados por entidades públicas ou privadas.

Como vimos, a prestação de um serviço poderá utilizar apenas parte das entidades da arquitetura. Por esta razão, o incremento de entidades na arquitetura não interfere significativamente com o desempenho da plataforma que a concretize, logo a arquitetura é escalável. Mais, a existência de entidades que forneçam serviços já existentes ou novos, poderá no mínimo manter o desempenho da plataforma. No entanto, é espetável que a inserção de novos serviços (mesmo que já existam entidades que os forneçam) venha a melhorar a qualidade do serviço prestado e mesmo o desempenho da plataforma que concretiza a arquitetura.

A segurança é sem sombra de dúvidas um dos aspectos mais críticos desta arquitetura; vamos analisar cada uma das seis propriedades identificadas previamente.

- Autenticação – esta propriedade é satisfeita recorrendo ao SIL com a autenticação do cidadão, que passa a informação para a entidade associada. A autenticação poderá ser realizada recorrendo a certificados emitidos por uma Entidade Certificadora (EC) da arquitetura para os requerentes. Esta informação é utilizada pelo entidade que tiver necessidade de o fazer, bastando para tal solicitar a informação à entidade que a armazena. Adicionalmente, entidades e repositórios de serviços são igualmente autenticados. Isto é conseguido no início da comunicação entre as duas entidades, através da troca de mensagens cifradas e assinadas digitalmente, que apenas podem ser interpretadas pelas entidades participantes.

- Autorização – Esta propriedade é atingida por duas componentes da entidade que providenciam avaliação da autorização. Uma dessas componentes verifica autorização de todos os atores envolvidos no processo até àquele momento; a outra verifica a autorização da entidade que requer a informação. As entidades devem respeitar as restrições de autorização antes de transferir os pedidos para o SIL ou processá-los de qualquer forma. Uma vez que cada SIL tem os seus próprios requisitos de Autorização, esta é apenas uma validação macroscópica. A micro validação deve ser feita pelo SIL que está associado, impondo os seus requisitos de segurança. A confiança é também validada. Apenas se os participantes num dado pedido de prestação de serviço respeitam as restrições de confiança impostas pelo serviço, a informação que produzem é de confiança. Outro ponto que merece referência é o facto de que o resultado de um determinado serviço não

circula em dados passados entre Entidades de um *workflow* a não ser que explicitamente pedido (*Result Request*) e o acesso aos dados apenas será facultado se o requerente for autorizado e confiável.

- Confidencialidade – como previamente mencionado, a comunicação entre duas entidades deve ser cifrada, o que garante a confidencialidade na troca de mensagens. No entanto, isto não garante que os agentes não possam aceder aos dados que não lhes sejam destinados. Para resolver esta questão, alguns itens das mensagens são também cifrados, o que é feito com recurso à chave pública da Entidade receptora.

- Rastreabilidade – como mencionado previamente, a rastreabilidade de um determinado serviço é garantida. No entanto, os dados relevantes estão espalhados pelas diversas entidades envolvidas na prestação do serviço. O carimbo temporal tem também um papel crítico no cumprimento do requisito, permitindo o mapeamento temporal de todas as mensagens que foram trocadas durante a prestação de um serviço.

- Integridade e Não-Repúdio – estes objetivos são assegurados por assinaturas digitais; cada mensagem está digitalmente assinada pelo seu emissor. Mais, há itens de mensagens que devem ser individualmente assinados. Por um lado a assinatura digital torna mais simples determinar se uma mensagem foi alterada, por outro lado garante que os dados foram gerados pela entidade que os assinou.

3.4 Conclusões

Neste capítulo descrevemos a arquitetura definida. É uma arquitetura de integração. A arquitetura utiliza o serviço de certificação para identificar, acreditar e autorizar entidades, para tal é necessário recorrer a uma estrutura de dados que designamos *Certificate*. Esta estrutura é utilizada para acreditar pares do tipo {Serviço, Entidade} e autorizar as entidades. As entidades comunicam entre si através da utilização de 4 tipos de mensagens padrão:

- *Request Service* – tem como principal objetivo a requisição de um serviço. É também através deste tipo de mensagem que é criada a composição de um serviço composto.

- *Notification* – tem como principal objetivo atualizar informação nas entidades. Mensagens deste tipo poderão conter informação sobre a aceitação de um serviço, sobre a produção de um resultado, sobre a ocorrência de um erro, sobre a alteração de estado de um serviço, etc.

- *Result Request* – este tipo de mensagens permitem o pedido de um resultado já produzido.

- *Result Delivery* – tem como único objetivo a devolução do resultado pedido.

Uma das principais preocupações durante a conceção da arquitetura foi a segurança. Como tal, um dos seus pressupostos é a manutenção dos resultados produzidos sob o controlo das entidades que os produziram, sendo apenas libertados mediante um pedido explícito. Este pequeno detalhe permite que as entidades supervisionem e decidam quem pode ter acesso a essa informação. Foi igualmente proposto um modelo de segurança, que se baseia no serviço de certificação e na emissão e revogação de credenciais, permitindo que as entidades que facultam serviços possam decidir qual o caminho que a prestação do serviço deve tomar.

Como demonstrámos, a arquitetura verifica todos os objetivos que identificámos inicialmente e atinge o objetivo a que nos propusemos. Esta arquitetura não tem limitações relativamente ao número de nós (entidades), nem ao número de serviços que poderá disponibilizar (sejam públicos, semi-públicos, ou privados). Para além disso, como é um *middleware* que permite a comunicação entre sistemas de informação das diferentes entidades, o utilizador final poderá usufruir da arquitetura a partir de qualquer canal que opte, desde que este esteja disponível e exista uma associação entre o respetivo SIL e a arquitetura. Isto é conseguido de forma transparente.

No capítulo seguinte iremos descrever o protótipo, os casos de utilização que foram escolhidos e as tecnologias utilizadas durante todo o processo de desenho e desenvolvimento.

Capítulo 4

Validação

No capítulo anterior identificámos um conjunto de objetivos para a criação de uma arquitetura de integração de serviços e apresentámos uma arquitetura que observa estes mesmos objetivos. Descrevemos, igualmente, um modelo de segurança que permite resolver as questões de segurança que identificámos.

Este capitulo tem como objetivo validar a arquitetura e o modelo proposto. Para tal, começaremos, na secção 4.1 por identificar um conjunto de casos de uso que abordam diferentes situações na AP e procedemos à sua descrição. Na secção 4.2 descrevemos uma implementação da arquitetura. Concluiremos o capítulo na secção 4.3.

4.1 Casos de uso

Esta secção tem como objetivo identificar os casos de uso a implementar de forma a validar as várias utilizações da arquitetura e a observância prática das suas características.

A validação da arquitetura será efetuada com base em quatro casos de uso: (a) Consulta médica; (b) Candidatura ao ensino superior; (c) Processo de permuta de imóveis; e, (d) Viagem ao estrangeiro. No caso (a) temos um serviço que é fornecido ao longo de um período relativamente longo de tempo, este caso tira total partido das capacidades de assincronismo da arquitetura. Isto deve-se essencialmente ao facto de haver necessidade da presença do paciente em locais distintos e ser necessário um período de tempo de espera para que os resultados possam ser gerados, para que o serviço seja concluído com sucesso. O caso (b) permite uma resolução bastante mais rápida do serviço, uma vez que em nenhum dos passos do processo existe necessidade de intervenção humana, permitindo, desta forma, a disponibilização dos resultados instantaneamente. O caso (c) representa o processo de permuta de imóveis realizado no Município de Portalegre. Este caso de uso foi obtido de um levantamento de processos efetuado no âmbito de um trabalho de investigação científica para a obtenção do grau de mestre. Este levantamento pode ser encontrado em [45]. Tal como no caso (b) este processo é concretizado de forma automática, sem existir necessidade de intervenção humana. No caso (d) temos um serviço que é disponibilizado por um organismo privado. Neste caso, para poder prestar o serviço ao seu cliente, a agência de viagens necessita de recorrer aos serviços do consulado de um país estrangeiro (relativamente ao cliente da agência de viagens). Neste caso, existe a necessidade de intervenção humana nos diferentes passos da prestação do serviço.

De notar que os casos de uso apresentados foram alterados de forma a permitirem a integração de serviços, não estando atualmente disponíveis tal como aqui se descrevem. Pretende-se,

com estes casos de uso, validar os seguintes pontos: integração de serviços; controlo de acesso ao serviço; controlo de acesso ao resultado; acreditação do serviço; acreditação do resultado; produção do resultado; e, autorização de participação no *workflow*.

4.1.1 Consulta Médica

Este caso de uso foi utilizado, na secção 3.3.1, para exemplificar o comportamento da arquitetura. Como pudemos observar na Figura 3.3, intervêm no processo três organismos (Clínica, Laboratório Clínico e Hospital) e quatro representantes (entidades) (Clínica, Laboratório Clínico, Hospital e Ministério da Saúde). De notar que o Ministério da Saúde é um organismo real. No entanto, no caso em questão não existe a necessidade de recorrer ao seu sistema de informação local (toda a informação para a prestação do serviço requisitado encontra-se acessível à entidade). Neste caso, a entidade tem como funções a decomposição do pedido do serviço, a solicitação dos pedidos resultantes da decomposição em serviços mais simples e agregação posterior desses serviços. Não necessita, por este motivo, de recorrer ao SIL do MS.

O caso em estudo é bastante simples: Um utente do sistema de saúde dirige-se ao seu médico de família para uma consulta de rotina. No entanto, o médico determina que existem razões para que o seu paciente seja observado por um médico especialista de forma a despistar eventuais problemas de saúde. O médico utiliza o seu sistema e solicita um conjunto de exames para o seu paciente e o reencaminhamento deste para uma consulta da especialidade no Hospital. Este processo recorre à entidade do Ministério da Saúde por ser aquele que tem a informação de como o serviço poderá ser decomposto. Seguem-se a marcação e realização de exames e marcação da consulta e observação do utente por parte do médico especialista.

Este serviço tem uma particularidade: com a exceção da intervenção protagonizada pelo agente do Ministério da Saúde, as intervenções para a prestação do serviço requerem a presença do paciente. Isto significa que, para além do desfasamento temporal normal, deste tipo de casos, entre o pedido inicial e a conclusão da prestação do serviço solicitado, a concretização da arquitetura não evita a deslocação do utente aos diferentes pontos da prestação do serviço. No entanto, simplifica todo o processo, uma vez que o utente beneficia da resolução de todo o processo burocrático, que é protagonizado pela concretização da arquitetura, ficando o utente apenas com a necessidade de comparecer para a prestação real do serviço.

Ainda com base neste caso de utilização implementámos uma sequência alternativa de eventos. Neste caso alternativo, ver Figura 4.1 (o diagrama de sequências não contempla todas as mensagens entre os agentes, apenas se encontram representadas as mensagens que podem afetar a execução do *workflow*), o processo é idêntico ao caso de utilização original até ao passo 6 (1.1.2 na Figura 4.1). Neste ponto, o LC 1 rejeita a prestação do serviço enviando uma mensagem do tipo *Notification*. A entidade do MS procura um novo prestador de serviço, encontrando-o. A partir deste ponto, o processo volta a ser idêntico ao processo original.

Este caso de uso utiliza os sete pontos em análise. Sempre que um serviço é solicitado, a entidade que solicita o serviço verifica se os destinatários do pedido estão acreditados para o fazer e se poderão fazer parte do *workflow*. Quando estes pedidos chegam aos seus destinatários estes verificam se quem os solicitou poderá fazer parte do *workflow* e se está autorizado a solicitar o serviço (a rejeição, por parte do Laboratório Clínico, em fornecer o serviço ao MS é um exemplo de uma não autorização). Relativamente ao pedido do resultado, o MS, antes de enviar o pedido, verifica se o resultado está acreditado e se foi produzido pela Laboratório Clínico. Por fim, quando o pedido do resultado chega ao Laboratório Clínico, este verifica se

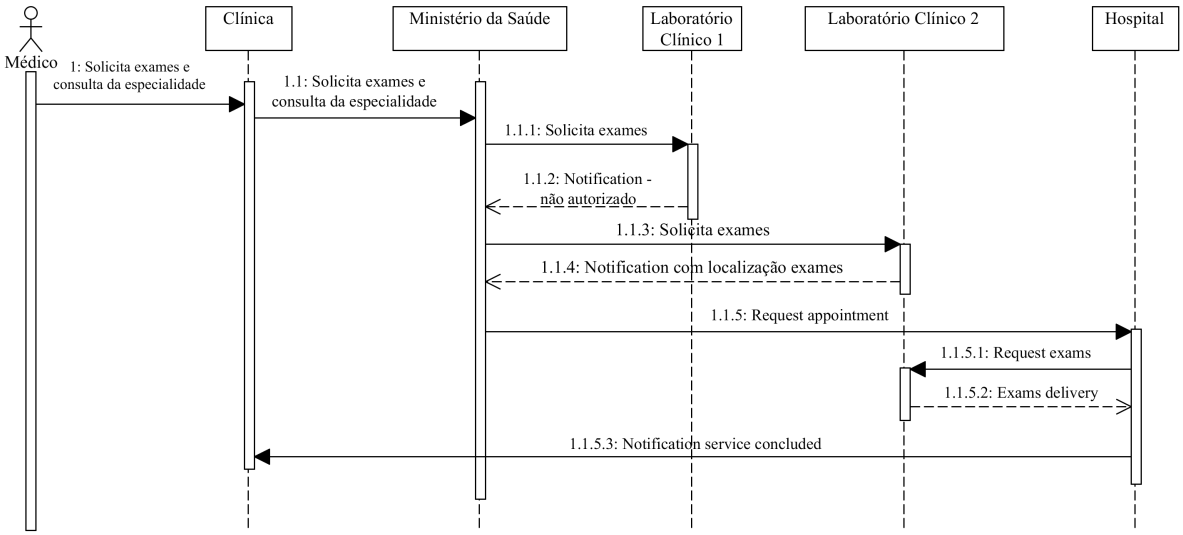

Figura 4.1: Caso de utilização Consulta médica, caso alternativo.

o Hospital (H) está autorizado a aceder ao resultado gerado pelo Laboratório Clínico e se está autorizado a participar no *workflow*.

4.1.2 Candidatura ao Ensino Superior

O caso de utilização Candidatura ao Ensino Superior, ao contrário do caso anterior, é totalmente automático a partir do momento em que é acionado pelo cliente. No entanto, este facto não influencia o normal decorrer de processos. A única diferença entre ambos os casos está relacionada com o tempo que decorre do pedido original do serviço até à conclusão da prestação do serviço. Este caso, tal como o anterior, é bastante simples e inicia-se quando um candidato ao ensino superior inicia o seu processo de candidatura. Esta candidatura realiza-se no SIL do Ministério da Educação e Ciência (MEC) que, por sua vez, tenta encontrar e utilizar a informação em falta: a nota do aluno (que poderá ser encontrada na escola de origem do aluno). Como podemos observar na Figura 4.2, temos duas entidades envolvidas (Ministério da Educação e Ciência e a Escola) e três representantes (Ministério da Educação e Ciência, Direção Geral da Administração Escolar e Escola) no processo. Tal como no exemplo anterior, a Direção Geral da Administração Escolar apenas se envolve no processo como agregador de serviços, identificando a localização da nota do candidato ao Ensino Superior.

Neste caso de uso, que está representado na Figura 4.2 (por simplicidade, o diagrama de sequências não contempla todas as iterações entre os agentes), o candidato acede ao Ministério da Educação e Ciência para fazer a sua candidatura ao ensino superior e seleciona os cursos e instituições para as quais se quer candidatar (1). Em seguida, o Sistema de Informação Local do MEC solicita as notas do candidato à entidade que lhe está associada. A entidade encontra o serviço da Direção Geral da Administração Escolar que pode fornecer as notas dos alunos, obtém a informação de que necessita para solicitar o serviço e solicita o serviço à entidade da Direção Geral da Administração Escolar (DGAE) (1.1). Por sua vez, a entidade da DGAE, não consegue responder ao serviço mas tem conhecimento da entidade que lhe poderá responder, reencaminha o pedido (ou parte dele) para a entidade da escola (1.1.1). A

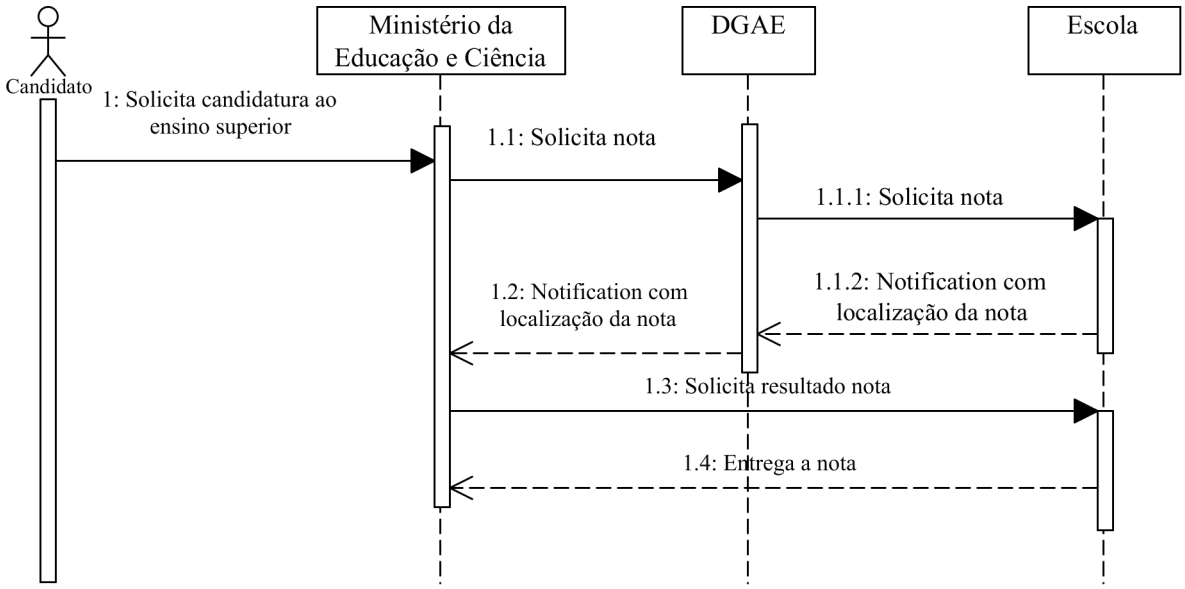

Figura 4.2: Diagrama UML de sequências que representa a Candidatura ao ensino superior.

entidade da escola solicita as notas ao SIL a que está associada e satisfaz o serviço enviando o endereço que identifica a localização das notas do aluno para a entidade da DGAE (1.1.2). A entidade da DGAE reencaminha a informação para a entidade do MEC (1.2). Uma vez que o MEC necessita do resultado para concluir o serviço inicialmente solicitado, envia um pedido para a entidade da escola a pedir o resultado do serviço (1.3). A satisfação deste último pedido conclui o serviço pedido originalmente (1.4). De notar que a entidade da escola, aquando da produção do resultado, não sabe quem será o destinatário final das notas do aluno. Esse destinatário só fica a ser conhecido quando a entidade do MEC solicita explicitamente o resultado produzido.

Como podemos constatar pela descrição do processo, o destinatário do resultado do pedido intermédio não é o requerente original, mas o prestador do serviço original. Neste caso em particular, a arquitetura evitou a deslocação do utente para adquirir toda a documentação necessária; e o utente pode fazê-lo a partir de um único lugar.

No caso alternativo que apresentamos o resultado final é igual, embora surjam duas entidades intermédias que fornecem o mesmo serviço, ver Figura 4.3. Ambas as entidades, Direção Regional de Educação do Norte (DREN) e Direção Regional de Educação do Centro (DREC) não têm acesso direto à nota do aluno, mas têm conhecimento de quem está habilitado a fazê-lo. Quer os passos iniciais, quer os passos finais deste caso são idênticos ao caso original. No entanto, supõe-se que a Escola não aceita o pedido proveniente da entidade representante da DREN (1.1.1) gerando por esta razão uma mensagem do tipo *Notification* a informar a entidade DREN (1.1.2). Por sua vez, a entidade DREN informa a entidade MEC que não conseguirá produzir o resultado esperado (uma alternativa a este passo seria o reencaminhar o pedido diretamente para a DREC). A partir deste momento o processo retoma a execução já descrita anteriormente. O MEC procura outro fornecedor do serviço, cria e envia um pedido de serviço. A candidatura ao Ensino superior é concluída quando a entidade MEC recebe a nota do aluno, podendo dar por terminado o processo.

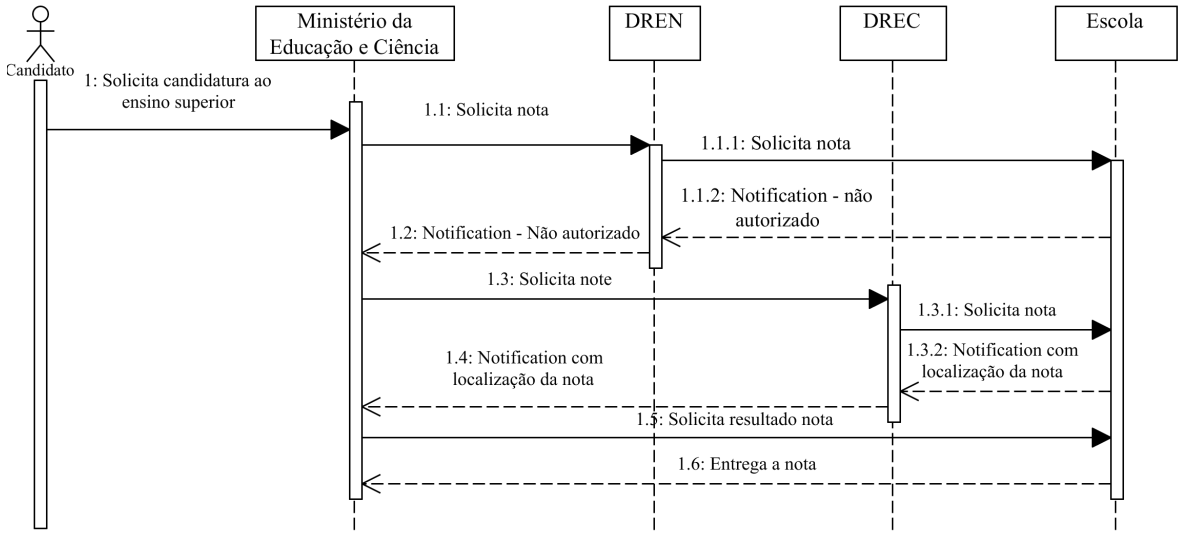

Figura 4.3: Diagrama UML de sequências que representa a Candidatura ao ensino superior, caso alternativo.

Este caso de uso utiliza os sete pontos em análise. Sempre que um serviço é solicitado, a entidade que solicita o serviço verifica se os destinatários do pedido estão acreditados para o fazer e se poderão fazer parte do *workflow*. Quando estes pedidos chegam aos seus destinatários estes verificam se quem os solicitou poderá fazer parte do *workflow* e se está autorizado a solicitar o serviço (a rejeição, por parte da Escola, em fornecer o serviço à DREN é um exemplo de uma não autorização). Relativamente ao pedido do resultado, o MEC, antes de enviar o pedido, verifica se o resultado está acreditado e se foi produzido pela Escola. Por fim, quando o pedido do resultado chega à Escola, esta verifica se o MEC está autorizado a aceder à nota e se está autorizado a participar no *workflow*.

4.1.3 Processo de permuta de imóveis

Neste caso de uso o utilizador, suponha-se, uma empresa, solicita a permuta de imóveis ao Município de Portalegre (MP). Para que esta permuta possa ser concretizada são necessários vários documentos. Estes documentos devem ser solicitados à Conservatória do Registo Predial de Portalegre (CRPP) e ao Serviço de Finanças de Portalegre (SFP) e correspondem a Certidão de teor do imóvel e Caderneta predial, respetivamente.

Neste caso de uso que podemos observar na Figura 4.4 o representante da Empresa acede ao MP para dar início ao processo de permuta (1). O SIL do MP solicita os documentos em falta à entidade MP que lhe está associada. A entidade MP encontra o serviço do SFP e envia-lhe o pedido de serviço (1.1). A entidade SFP aceita o pedido e reencaminha-o para o seu SIL, obtendo a resposta com o documento em seguida e a entidade do SFP envia uma mensagem do tipo *Notification* a informar a entidade do MP sobre a localização do documento gerado (1.2). Em seguida, a entidade MP solicita explicitamente o documento (1.3), que lhe é enviado (1.4). Por fim, a entidade MP procura por entidades que gerem a Caderneta Predial e encontra a entidade CRPP a quem envia uma mensagem do tipo *Service Request* a solicitar o documento (1.5). De notar que neste caso, nada impede que as solicitações ao SFP e ao

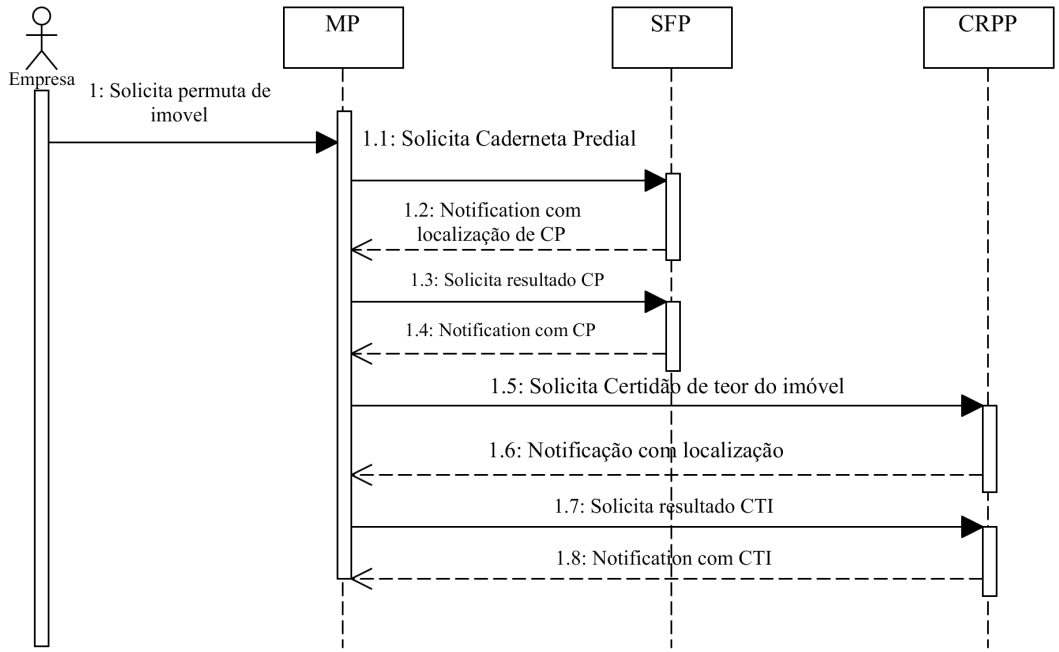

Figura 4.4: Caso de utilização Processo de permuta.

CRPP sejam realizadas em simultâneo. Por sua vez a entidade do CRPP recebe o pedido de serviço, solicita a prestação do serviço ao seu SIL e recebe o resultado. A entidade CRPP envia para a entidade MP com a localização do documento produzido (1.6). O processo é concluído com a solicitação explicita do resultado (1.7) e seu envio (1.8).

Este caso de uso utiliza os sete pontos em análise. Sempre que um serviço é solicitado, a entidade que solicita o serviço verifica se os destinatários do pedido estão acreditados para o fazer e se poderão fazer parte do *workflow*. Quando estes pedidos chegam aos seus destinatários estes verificam se quem os solicitou poderá fazer parte do *workflow* e se está autorizado a solicitar o serviço. Relativamente ao pedido do resultado, o MP, antes de enviar o pedido, verifica se o resultado está acreditado e se foi produzido pela entidade. Por fim, quando o pedido do resultado chega ao SFP, este verifica se o MP está autorizado a aceder ao documento e se está autorizado a participar no *workflow*.

4.1.4 Viagem ao estrangeiro

O caso de utilização é bastante simples: um cidadão precisa de viajar para outro país e solicita a uma agência de viagens a compra do bilhete, a reserva do alojamento e o pedido do visto de entrada (ver Figura 4.5). Apenas o pedido do visto está contemplado no caso de uso. Apenas as trocas de mensagens entre as entidades estão representadas no diagrama de sequências.

Após a receção do pedido efetuado pelo seu cliente (1), a agência de viagens envia um pedido para a emissão do visto ao Consulado do país para onde o seu cliente pretende viajar (1.1). Após a emissão do visto o Consulado envia uma mensagem do tipo Notification - Result Identifier com a informação sobre a localização do resultado (1.2). Em seguida, a agência de

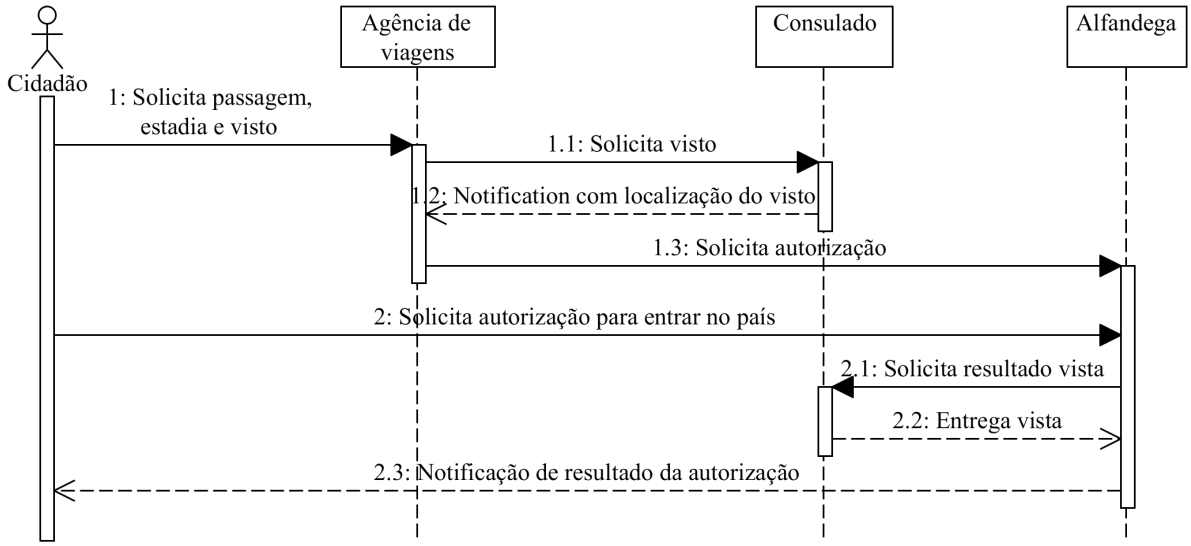

Figura 4.5: Diagrama UML de sequências que representa a aquisição e apresentação do visto de entrada num país estrangeiro.

viagens envia a informação à Alfandega do país de destino como um pedido para admitir o seu cliente no país (1.3).

O cidadão, quando chega ao país de destino solicita a entrada (2). A Alfandega já está em poder da localização do Visto de entrada e solicita-o ao Consulado que o emitiu (2.1). O Consulado, envia o Visto para a Alfandega (2.2). A Alfandega permite a entrada do cidadão (2.3).

O caso de uso exemplifica a utilização de uma concretização da arquitetura para simplificar a interação entre entidades públicas e privadas. Neste caso em concreto, os intervenientes no *workflow* são de diferentes países.

Este caso de uso demonstra o potencial da arquitetura, uma vez que não limita as entidades que podem solicitar ou fornecer serviços, desde que estejam devidamente credenciadas para o efeito. Salientamos o facto de o visto apenas ter saído do consulado quando é solicitado pela Alfandega.

Este caso de uso utiliza os sete pontos em análise. Para solicitar os serviços "Pedido de visto"e "Pedido de entrada no país"a agência de viagens necessita de verificar se o Consulado e a Alfândega estão acreditados para realizar os serviços que anunciam e se poderão fazer parte do *workflow*. Quando estes pedidos chegam aos seus destinatários estes verificam se a agência poderá fazer parte do *workflow* e se está autorizada a solicitar o serviço. Relativamente ao pedido do resultado, a Alfândega, antes de enviar o pedido, verifica se o resultado está acreditado e se foi produzido pelo Consulado. Por fim, quando o pedido do resultado chega ao Consulado, este verifica se a Alfândega está autorizada para aceder ao resultado e se está autorizada para participar no *workflow*.

A arquitetura tem um comportamento semelhante em qualquer das situações apresentadas, independentemente de serem serviços em que estejam envolvidos cidadãos, empresas ou

instituições públicas, e independentemente de se tratarem de serviços com prestação síncrona ou assíncrona.

4.2 Implementação da arquitetura

Nesta secção vamos abordar as questões relacionadas com a implementação da arquitetura. Na secção 4.2.1 identificamos a plataforma de agentes utilizada no desenvolvimento do protótipo e concluímos com a descrição da implementação das componentes da arquitetura na secção 4.2.2.

4.2.1 Agentes de *software*

Existem diversos tipos de agentes de *software*: a classificação depende da sua utilização. O espectro de utilização dos agentes é bastante alargado: podem ser utilizados em interfaces de sistemas operativos, processamento de imagens de satélites, em gestão distribuída de eletricidade, controlo aéreo, gestão de processos de negócios, comércio eletrónico, jogos de computadores, etc. [41].

O núcleo da arquitetura é o conceito entidade (ver secção 3.3.1). Optámos por implementar este conceito recorrendo a agentes de *software*, devido ao seu conjunto de características intrínsecas. Agentes de *software* são definidos como componentes de *software* que são reativas (agem em resposta a um dado evento), proativas (adaptam-se às alterações do seu meio ambiente), autónomas (são capazes de atuar sem intervenção exterior) e são capazes de comunicar entre si [41, 28]. Estas características encontram-se igualmente no conceito entidade, por esta razão e pelo facto de os agentes de *software* serem adequados para a computação distribuída e para a composição de serviços [56]

Nesta secção indicamos as razões que suportam a nossa decisão para implementar a arquitetura com base numa plataforma de agente e apresentamos a plataforma de agentes que serviu de suporte ao desenvolvimento do protótipo. Na secção 4.2.1 apresentamos a plataforma de agentes que utilizamos, justificamos a sua escolha e fundamentamos a utilização da extensão à plataforma que permite a execução de *workflows*. Concluímos a secção com uma breve descrição da extensão WADE.

Java Agent DEvelopment Framework

Para desenvolver o protótipo optámos por utilizar a plataforma de agentes JADE [8, 63]. O JADE é uma plataforma desenvolvida em Java que segue as especificações da *Foundation for Intelligent, Physical Agents* (FIPA). Esta plataforma tem um conjunto de ferramentas e bibliotecas que permitem o rápido desenvolvimento de aplicações para diversos tipos de equipamentos e com diversas finalidades.

A escolha recaiu sobre esta plataforma uma vez que é uma das plataformas de agentes mais utilizada (quer a nível académico, quer a nível empresarial), tendo uma comunidade bastante forte e ativa. A sua versatilidade e robustez podem ser demonstradas pela variedade de aplicações que a utilizam.

O JADE tem um conjunto de complementos que adicionam funcionalidades extra à plataforma. Um destes exemplo é o *Web Service Integration Gateway* (WSIG). Este complemento permite a utilização de Serviços Web pelos Agentes. Uma das principais razões para optar pelo JADE, é a existência de uma extensão à plataforma que permite a implementação de

workflows por agentes WADE [64], o que nos permite uma maior aproximação do protótipo implementado ao desenho efetuado e já descrito no capítulo 3. Outro dos pontos que nos fez optar por esta plataforma de agentes foi a sua documentação, a rapidez de aprendizagem dos conceitos base necessários e a rapidez de implementação.

Workflows and Agent Development Environment

O WADE é uma extensão à plataforma de agentes JADE e permite a definição e execução de tarefas por parte dos agentes. O conjunto destas tarefas perfazem um *workflow*, que é utilizado na implementação com o objetivo de estabelecer o comportamento das entidades na arquitetura. A arquitetura que apresentámos na secção 3.3 utiliza este conceito. Outra das vantagens (embora não tenhamos tirado partido dela) é o facto de o complemento WSIG estar ativo por omissão. Tanto o WADE como o JADE utilizam a linguagem Java, permitindo a utilização do paradigma orientado a objetos e tirar partido das vantagens deste paradigma.

4.2.2 Componentes

Como exposto na secção 4.2.1, o protótipo que concretiza a arquitetura foi desenvolvido recorrendo ao JADE e à extensão WADE. O protótipo foi implementado em Java tendo respeitado o desenho apresentado no capítulo 3. A implementação está organizada segundo a estrutura representada no diagrama da Figura 4.6. No entanto, uma vez que o objetivo principal deste protótipo é aferir a validade do modelo no que concerne ao controlo de acesso aos serviços, ao controlo de acesso aos resultados, à acreditação dos resultados e dos serviços, à produção dos resultados e à participação no *workflow*, apenas foram implementadas as estruturas fundamentais para o efeito.

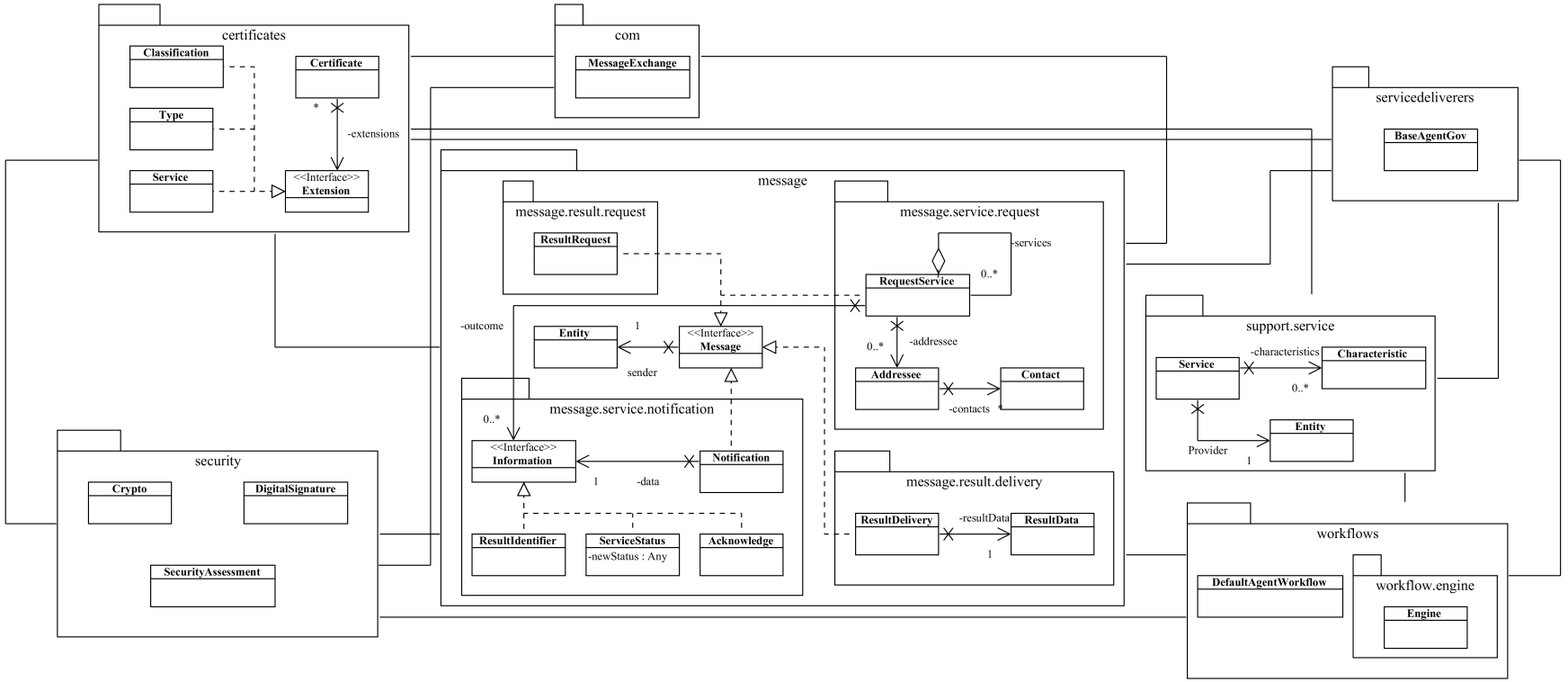

Figura 4.6: Diagrama UML de pacotes da implementação.

Nas restantes secções iremos descrever como foram implementadas as diversas componentes da arquitetura.

Serviços de suporte

Como descrito na secção 3.3.3 a arquitetura proposta utiliza três estruturas de suporte, relembrando: Serviço de carimbo temporal - para estabelecer a data e hora da criação e envio de uma mensagem permitindo a ordenação cronológica das mensagens que são geradas durante a prestação de um serviço; Serviço de Diretoria - que permite o registo e a pesquisa dos serviços disponibilizados pelas entidades; e o Serviço de Certificação - que estabelece, gere e atribuí os certificados que definem quais os serviços que uma entidade está acreditada para prestar ou quais aqueles que, de acordo com a classificação que lhe foi atribuída, poderá solicitar.

Qualquer um destes serviços não é essencial para atingir o objetivo pretendido com o protótipo. A não atribuição de uma data e uma hora (por uma entidade acreditada para o efeito) à criação da mensagem não invalida a utilização da mensagem e o seu reconhecimento pelos restantes intervenientes no processo.

O serviço de diretoria é, igualmente, uma componente que pode ser omitida, embora reconheçamos o seu valor no compto geral da arquitetura. É esta componente que faculta os dados que servem para realizar a escolha de qual o prestador a executar o serviço solicitado. No entanto, este fator não é essencial para o protótipo uma vez que para a prova de conceito apenas necessitamos que cada entidade tenha conhecimento dos serviços que estão disponíveis. Esse conhecimento foi diretamente introduzido nos agentes implementados.

Relativamente aos serviços de certificação, os certificados emitidos por estes serviços permitem que as entidades que utilizam a arquitetura possam fornecer ou consumir serviços de acordo com os atributos que são inseridos nestes certificados. Uma vez que no protótipo não implementamos este serviço, assumimos que qualquer certificado é emitido por uma entidade certificadora acreditada para o efeito e que, excetuando o caso da validade temporal (que é verificada), os restantes atributos dos certificados são considerados válidos.

Embora os serviços de suporte sejam uma parte integrante e fulcral da arquitetura a sua omissão na implementação da arquitetura não impede que se validem os objetivos, desde que se garantam os pressupostos, como é o caso. Assim, as simplificações usadas não comprometem a prova de conceito que pretendemos realizar com o protótipo desenvolvido.

Estruturas de suporte

A estrutura *Certificate* (ver secção 3.3.3) apenas foi implementada parcialmente. Esta estrutura está contida no pacote *certificates* (ver Figuras 4.6 e 4.7). Todas as classes identificadas no desenho da arquitetura foram implementadas(ver secção 3.3.3) no protótipo. No entanto, apenas foram utilizadas as classes e atributos essenciais para a identificação, acreditação e autorização, de forma a ser possível atingir os nossos objetivos para a validação.

Os atributos implementados foram os seguintes: *version*; *issuer*; *subject*; e *validity*. E as classes que realizam a interface *Extension* (*Classification*, *Type*, *Service*). Este conjunto de campos permite identificar os intervenientes no processo (*subject*), verificar a validade do certificado (através do intervalo temporal da sua validade - *validity*) e verificar o caminho de certificação (*issuer*). Esta implementação parcial permite-nos manter as virtudes do conceito, por um lado, e, por outro, simplificar todo o processo de prestação de serviços de forma a demonstrar o conceito.

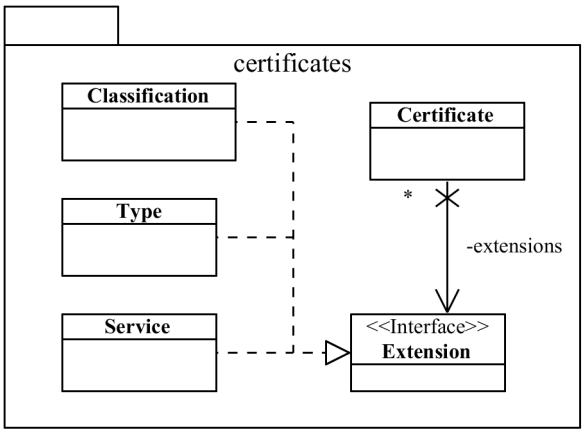

Figura 4.7: Diagrama de classes da estrutura Certificate.

Por fim, as classes *Classification*, *Type* e *Service* foram também implementadas. Isto permite armazenar a informação referente à acreditação e à autorização de um agente relativamente a um serviço, que possa fornecer ou consumir, respetivamente.

De notar que não utilizamos os campos referentes à chave pública e à assinatura. A não utilização destes campos não nos permite comprovar a assinatura digital nem se o certificado foi emitido pela EC por quem está indicado no campo *issuer*. No entanto, qualquer destas funcionalidades está amplamente divulgada e testada, pelo que a sua inclusão não traria valor acrescentado à prova de conceito e, tal como indicámos, pressupomos que todos os certificados utilizados são emitidos por uma EC para tal.

A estrutura *Service Description*, também descrita na subsecção 3.3.3, cujas instâncias armazenam a descrição de um serviço publicado por um prestador de serviços, foi totalmente implementada (ver Figura 4.8). Está acessível na arquitetura na package *support.service* (ver Figuras 4.6 e 4.8). Embora esta estrutura tenha sido implementada, apenas a classe *Provider* e o atributo *description* da classe *Service* são utilizados durante a execução dos casos de utilização descritos. Esta escolha deve-se ao facto dos atributos utilizados serem essenciais para a descrição do serviço e para a identificação e localização de quem os fornece.

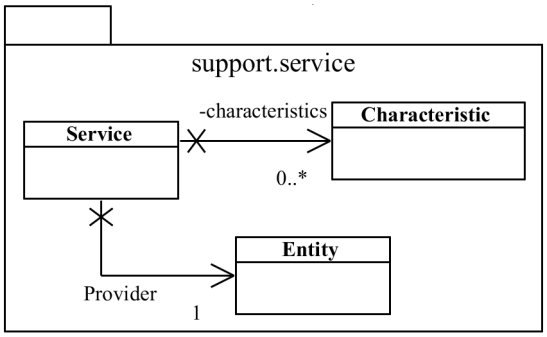

Figura 4.8: Diagrama UML de classes da estrutura Service implementada.

Na secção anterior, referimos que os serviços de suporte não foram implementados, o que

inclui naturalmente, o serviço de diretório. No entanto, é essencial que a informação sobre os serviços exista e esteja acessível para os agentes. As instâncias da estrutura *Service* são criadas e estão disponíveis como atributos nos agentes que necessitam da informação.

Mensagens

A estrutura das mensagens (ver secção 3.3.3) foi implementada na totalidade e de acordo com a Figura 4.9. Isto permite que a comunicação entre agentes seja desenvolvida e que os agentes possam executar o seu processo normal com base na mensagem que recebem. As estruturas foram implementadas no pacote *message* (ver Figuras 4.6 e 4.9).

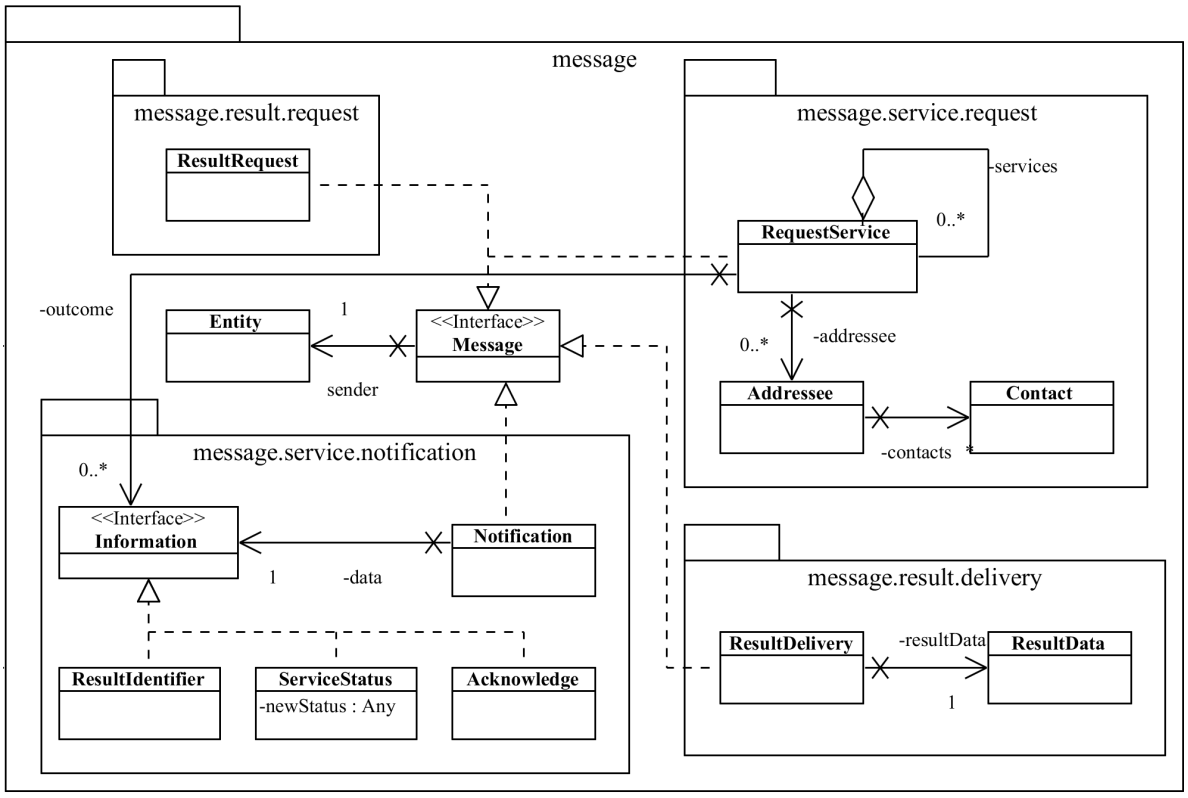

Figura 4.9: Diagrama de classes da estrutura Message no protótipo.

Tal como mencionámos anteriormente, o par de chaves que dá suporte à assinatura digital e aos mecanismos de cifra não foi gerado. Adicionalmente, as funções criptográficas não foram implementadas. Em suma, no protótipo desenvolvido os mecanismos para a assinatura digital e cifra e decifra não foram implementados. No entanto, estes processos são conhecidos e sem implicações para a prova de conceito que pretendemos efetuar.

Agentes

Uma das componentes principais da arquitetura são as entidades que são implementadas recorrendo a agentes de *software*. A implementação dos agentes segue o comportamento descrito na secção 3.3.3. As classes que implementam o agente genérico estão distribuídas

por quatro pacotes: *servicedeliverers* - pacote que contém a classe que representa um agente genérico; *workflow* - pacote que contém a classe onde se encontram definidas as tarefas a realizar por um agente; *com* - neste pacote estão definidas as classes que permitem a um agente preparar uma mensagem para envio (assinando digitalmente a mensagem, cifrando, etc), onde se encontram as funções que permitem a receção de mensagens e o tratamento das mensagens recebidas (decifrando a mensagem recebida, verificando assinaturas, etc); por fim, o pacote *security* - contém todas as classes que possibilitam a verificação dos pressupostos de segurança e as funções de criptografia.

Qualquer agente que utilize a extensão WADE da plataforma JADE é um subtipo da superclasse *WorkflowEngineAgent*. No protótipo o pacote *servicedeliverers* (ver Figura 4.10) contém a representação genérica de um agente da arquitetura, a classe *BaseAgentGov*. É nesta classe que estão definidos os comportamentos do agente relativamente à rede onde se insere (comunicação com o SIL, envio e receção de mensagens para e de os restantes agentes). No protótipo, todas as classes que representem entidades são subclasses da classe *BaseAgentGov* e é nestas classes que se instanciam os certificados, os serviços (relembramos que os repositórios de serviços não foram implementados, pelo que foi necessário inserir esta informação diretamente na instanciação dos agentes) e o *workflow* a ser executado (o *Service Workflow Warehouse* - ver Figura 3.14 - não se encontra implementado e optámos por inserir a informação sobre o *workflow* na instanciação do agente).

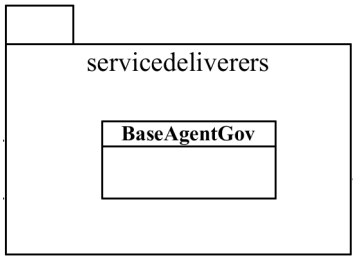

Figura 4.10: Diagrama UML de classes do pacote *servicedeliverers*.

A classe que define o *workflow* interno do agente denomina-se por *DefaultAgentWorkflow* (ver Figura 4.11). Nesta classe está definido o conjunto de tarefas que um agente deve executar após a receção de uma mensagem (a sequência de tarefas a executar depende do tipo de mensagem recebida). A classe *DefaultAgentWorkflow* utiliza como suporte a classe *Engine* da package *workflow.engine*. A classe *Engine* destina-se a conter o processo para gerar (não implementado) e gerir a parte do *workflow* que pertence ao agente a quem foi solicitada a prestação do serviço, exercendo o controlo sobre a parte do *workflow* que conhece (ver Figura 3.14, atividades *Generate Workflow* e *Manage Workflow*).

A classe *MessageExchange* (ver Figura 4.12) contém as funcionalidades que permitem preparar uma mensagem para envio, assinando a mensagem, solicitando o carimbo temporal para a mensagem, cifrando-a, etc. e as funcionalidades que permitem fazer o processo inverso. Ou seja, após a receção da mensagem proceder de forma a que seja possível aceder ao seu conteúdo, decifrando a mensagem, verificando a sua assinatura e o carimbo temporal, etc.

Por fim, a *package security* contém as classes com a implementação das funcionalidades relacionadas com a segurança. A classe *Crypto* (não foi implementada) contém as funções que lidam com a cifra e a decifra de mensagens. A classe *DigitalSignature* (não implementada) de-

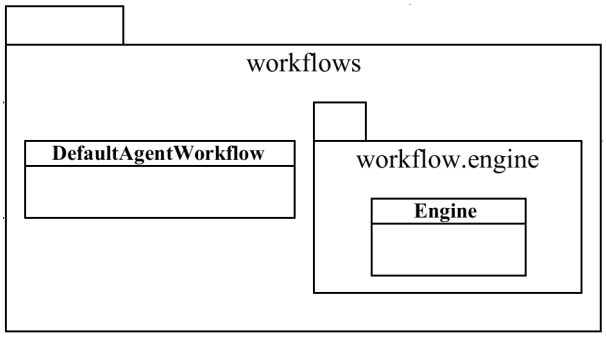

Figura 4.11: Diagrama UML de classes do pacote *workflows*.

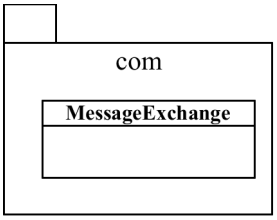

Figura 4.12: Diagrama de classes da package *com*.

verá conter todas as funcionalidades relacionadas com a assinatura digital e com a verificação da assinatura digital. Por fim, a classe *SecurityAssessment* contém todas as funções relacionadas com a autorização e acreditação e que têm um impacto direto com o protótipo (*Workflow Authorization*, *Assessment Service Authorization*, *Assessment Result Authorization*, *Result Accreditation*, *Service Accreditation* e *Result Production*)

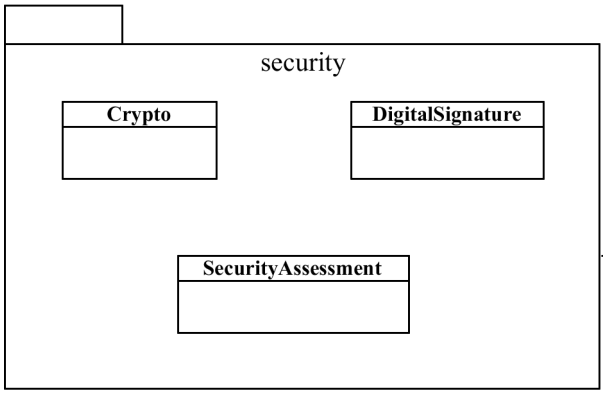

Figura 4.13: Diagrama UML de classes da package *security*.

4.2.3 Resultado

Durante a execução dos casos de utilização, cada um dos agentes que intervem na prestação do serviço regista a sua atividade. Nesta secção vamos mostrar e clarificar alguns pontos (correspondentes a uma iteração do agente) do registo efetuado. Vamos apenas descrever os pontos mais importantes do *workflow* interno do agente, as restantes linhas do registo pertencem apenas a passos intermédios. A iteração que vamos analisar corresponde à receção do pedido de serviço por parte do agente da clínica (ver Figura 3.2 referente ao caso de utilização apresentado em secção 4.1.1).

5	BaseGovAgent:	clinic1 AgentMessageReception SERVICEREQUEST from LIS
6	DefaultAgentWorkflow:	———————— Starting workflow clinic1 ————————
7	DefaultAgentWorkflow:	clinic1 MessageReceptionExecution Starting SERVICEREQUEST

Figura 4.14: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 1.

Na Figura 4.14 encontra-se o registo da receção, por parte do agente, de uma mensagem do tipo *Request Service* proveniente do seu SIL (5).

A mensagem recebida é enviada para processamento interno (ver Figura 3.14, atividade *Receive Message* e Figura 4.15 (8), onde é decifrada (9), o certificado que identifica o remetente é verificado (10), a assinatura digital é verificada (11). Uma vez que a verificação efetuada foi positiva a mensagem pode ser processada (12). Após a conclusão deste processo inicial, o tipo da mensagem está determinado, é um *Request Service*.

8	MessageExchange:	MessageReception Reception ServiceRequest
9	MessageExchange:	MessageReception Deciphering SERVICEREQUEST
10	MessageExchange:	CertificateValidation Valid SERVICEREQUEST
11	MessageExchange:	DigitalSignatureVerification Verified SERVICEREQUEST
12	MessageExchange:	MessageProcessing CanBeProcessed SERVICEREQUEST

Figura 4.15: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 2.

O tipo da mensagem impõe um caminho ao *workflow* interno do agente. Neste caso, é necessário determinar se o agente que solicitou o serviço está autorizado para o fazer e se está autorizado a participar no *workflow*. O pedido vai ser analisado com base nos certificados do agente requerente (ver Figura 3.14, atividade *Assess Service Authorization* e Figura 4.16)(16). Neste ponto são obtidos e validados os certificados do solicitador do serviço, para além de ser determinado se o solicitador do serviço está autorizado a aceder ao serviço (17, 18). Em seguida são determinados todos os intervenientes no *workflow* do serviço (19) e a sua autorização de participação no *workflow* é aferida (*BuilBackwardPath*) (20).

Após a verificação das credenciais e de ter sido facultado acesso ao *workflow* e à prestação do serviço, o serviço solicitado é decomposto (24) (ver Figura 3.14, atividade *Generate workflow* e Figura 4.17) e o *workflow* é adicionado à *Workflow Warehouse* (25).

Este pedido dá origem a duas mensagens: uma mensagem do tipo *Notification* (26, 27, 28) (ver Figura 4.18) e uma mensagem do tipo *Request Service*.

13	DefaultAgentWorkflow:	clinic1 MessageReceptionExecution Received SERVICEREQUEST
14	DefaultAgentWorkflow:	clinic1 AssessingServiceAuthorization Starting SERVICEREQUEST
15	DefaultAgentWorkflow:	clinic1 AssessingServiceAuthorization Assessing SERVICEREQUEST
16	SecurityAssessment:	AssessingServiceAuthorization Starting SERVICEREQUEST
17	SecurityAssessment:	AssessingServiceAuthorization GettingCertificates SERVICEREQUEST
18	SecurityAssessment:	AssessingServiceAuthorization Executing SERVICEREQUEST
19	SecurityAssessment:	AssessingServiceAuthorization BackwardWorkflowPath SERVICEREQUEST
20	SecurityAssessment:	AssessingServiceAuthorization AssessingAuthorization SERVICEREQUEST

Figura 4.16: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 3.

21	SecurityAssessment:	AssessingServiceAuthorization Assessed SERVICEREQUEST
22	Engine:	————- Starting Workflow Generation .... ————-
23	Engine:	WorkflowGeneration Starting SERVICEREQUEST
24	Engine:	WorkflowGeneration Decomposition SERVICEREQUEST
25	Engine:	WorkflowGeneration AddingToWorkflowWarehouse SERVICEREQUEST

Figura 4.17: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 4.

26	Engine:	WorkflowGeneration NotificationCreation SERVICEREQUEST
27	Engine:	WorkflowGeneration NotificationCreated NOTIFICATION
28	Engine:	WorkflowGeneration Ending NOTIFICATION

Figura 4.18: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 5.

Durante a execução da atividade *Manage Workflow*, a informação sobre o serviço é atualizada no *Workflow Warehouse* (30, 31, 32) (ver Figura 4.19).

29	DefaultAgentWorkflow:	clinic1 WorkflowManagement Starting SERVICEREQUEST
30	Engine:	WorkflowManagement Starting SERVICEREQUEST
31	Engine:	WorkflowManagement Updating SERVICEREQUEST
32	Engine:	Workflow UpdatingWorkflowWarehouse SERVICEREQUEST

Figura 4.19: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 6.

Em seguida é determinado o próximo passo a executar (34) e são estabelecidos os parâmetros a utilizar na escolha do próximo prestador de serviços (35, 36, 37, 38, 39)(ver Figura 4.20).

A conclusão deste processo passa pela verificação da acreditação do agente a quem será

33	Engine:	Workflow AnalyzingWorkflowStatus SERVICEREQUEST
34	Engine:	Workflow GettingNextTask sr.getType()
35	Engine:	WorkflowDelegation GettingCharacteristics SERVICEREQUEST
36	Engine:	WorkflowDelegation OrderingProviders SERVICEREQUEST
37	Engine:	WorkflowDelegation ChoosingProvider SERVICEREQUEST
38	Engine:	ChoosingProvider Starting sr.getType()
39	Engine:	ChoosingProvider GettingCertificates sr.getType()

Figura 4.20: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 7.

delegado o seguimento do processo. Se as condições para poder delegar o controlo da prestação do serviço se verificarem a prestação do serviço prossegue com o envio de um pedido de serviço para o agente destinatário (40, 41, 42)(ver Figura 4.21).

40	SecurityAssessment:	AssessingServiceAccreditation Starting rd.getType()
41	SecurityAssessment:	AssessingServiceAccreditation CheckingValidity rd.getType()
42	SecurityAssessment:	AssessingServiceAccreditation CheckingCertificatePath rd.getType()

Figura 4.21: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 8.

O último passo a dar é o envio das mensagens para os seus destinatários. Para tal é necessário preparar a mensagem para envio (ver Figura 4.22): assinando (48, 55); adicionando o carimbo temporal (49, 56) e cifrando a mensagem (50, 57).

Após a conclusão do *workflow* interno do agente as mensagens são enviadas para os seus destinatários (64, 68) (ver Figura 4.23).

4.3 Considerações

Neste capítulo apresentámos o protótipo que foi desenvolvido com base no desenho descrito no Capítulo 3, na plataforma de agentes JADE e na extensão WADE.

O protótipo descrito é apenas uma possível solução para a arquitetura proposta; outras soluções poderão ser implementadas. No protótipo foram implementadas todas as estruturas de suporte de dados, embora apenas parte destas estruturas tenham sido efetivamente utilizadas na implementação e execução dos casos de utilização. Embora todas as estruturas de dados tenham sido implementadas, os serviços de certificação, de carimbo temporal e o repositório de serviços não foram implementados. Argumentamos que estes serviços não são essenciais para a prova de conceito da arquitetura.

Foram implementados 4 casos de utilização: Consulta médica; Candidatura ao ensino superior; Permuta de imóveis; Viagem ao estrangeiro. Para os primeiros dois casos foram ainda implementados casos alternativos onde uma das entidades, a quem tinha sido solicitado um serviço, rejeitava participar no processo.

Da execução dos casos de utilização obtivemos o registo efetuado pelo protótipo e que nos permitiu validar o processo interno do agente e fazer uma análise ao comportamento do protótipo.

43	Engine:	ChoosingProvider PreparingMessage sr.getType()
44	DefaultAgentWorkflow:	clinic1 WorkflowManagement Ended SERVICEREQUEST
45	DefaultAgentWorkflow:	clinic1 MessageSending Starting
46	DefaultAgentWorkflow:	clinic1 MessageSending GettingMessage NOTIFICATION
47	DefaultAgentWorkflow:	clinic1 MessageSending PreparingMessage NOTIFICATION
48	MessageExchange:	SigningMessage Signed NOTIFICATION
49	MessageExchange:	MessageTimestamping TimeStamped NOTIFICATION
50	MessageExchange:	MessageEncryption Encrypted NOTIFICATION
51	DefaultAgentWorkflow:	clinic1 MessageSending Encrypting NOTIFICATION
52	DefaultAgentWorkflow:	clinic1 MessageSending Sending NOTIFICATION
53	DefaultAgentWorkflow:	clinic1 MessageSending GettingMessage SERVICEREQUEST
54	DefaultAgentWorkflow:	clinic1 MessageSending PreparingMessage SERVICEREQUEST
55	MessageExchange:	SigningMessage Signed SERVICEREQUEST
56	MessageExchange:	MessageTimestamping TimeStamped SERVICEREQUEST
57	MessageExchange:	MessageEncryption Encrypted SERVICEREQUEST
58	DefaultAgentWorkflow:	clinic1 MessageSending Encrypting SERVICEREQUEST
59	DefaultAgentWorkflow:	clinic1 MessageSending Sending SERVICEREQUEST
60	DefaultAgentWorkflow:	clinic1 Workflow Ended
61	DefaultAgentWorkflow:	———————— Ending workflow clinic1 ————————
62	BaseGovAgent:	clinic1 AgentWorkflowConclusion Succeeded null

Figura 4.22: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 9.

63	BaseGovAgent:	clinic1 AgentWorkflowDelegation SendMessage NOTIFICATION
64	BaseGovAgent:	clinic1 AgentWorkflowDelegation SendMessage Addressee: LIS
65	BaseGovAgent:	clinic1 AgentWorkflowDelegation MessageSent Addressee: LIS
66	BaseGovAgent:	clinic1 AgentWorkflowDelegation SendMessage SERVICEREQUEST
67	BaseGovAgent:	clinic1 AgentWorkflowDelegation SendMessage Addressee: ministerio
68	BaseGovAgent:	clinic1 AgentWorkflowDelegation MessageSent Addressee: ministerio

Figura 4.23: Registo de uma iteração do caso de utilização *Consulta Médica* - Parte 10.

A arquitetura proposta no capítulo anterior foi validada com base nos casos de uso identificados e no protótipo implementado. Para a validação foram consideradas: a integração de serviços; a autorização de participação no *workflow*; a acreditação do serviço; a acreditação do resultado; a autorização de acesso ao serviço; e, a autorização de acesso ao resultado. Podemos por esta razão concluir que as entidades têm os mecanismos de segurança adequados para controlar a sua participação, a participação de terceiros e a privacidade dos resultados produzidos no âmbito dos mesmos.

Capítulo 5

Conclusões

Ao longo do documento, nos quatro capítulos anteriores: identificámos o problema, apresentámos a motivação, os objetivos, as contribuições e as publicações produzidas no âmbito do trabalho; identificámos os quadros de referência para a interoperabilidade que são relevantes para este trabalho, os paradigmas arquiteturais utilizados, analisámos um conjunto de arquiteturas e/ou plataformas de integração relativamente à criação de *workflows* dinâmicos e às suas soluções de segurança, e um conjunto de modelos de controlo de acesso que podem ser utilizados na arquitetura que propomos; apresentámos uma nova arquitetura de integração que permite a construção dinâmica de *workflows* e permite que os participantes decidam sobre a sua participação, sobre a quem vão delegar o serviço e a quem entregam o resultado previamente produzido; e descrevemos um conjunto de casos de uso que implementámos num protótipo que desenvolvemos para o efeito. Neste capítulo, concluímos a dissertação fazendo uma resenha dos principais resultados e identificando alguns pontos para trabalho futuro.

5.1 Principais resultados

Nesta secção, apresentamos de forma sucinta os principais resultados do trabalho de doutoramento. O trabalho suporta-se sobre três resultados:

- Levantamento das questões de segurança relacionadas com a construção de *workflows* dinâmicos no âmbito da integração de serviços no governo eletrónico;
- Definição de um conjunto de princípio e de regras (arquitetura) que permita resolver as questões de segurança levantadas pela construção de *workflows* dinâmicos no governo eletrónico;
- Validação da arquitetura recorrendo à identificação de casos de uso que exemplifiquem o uso de *workflows* construídos dinamicamente e implementação destes num protótipo desenvolvido com este propósito.

5.1.1 Levantamento das questões de segurança

A integração de serviços na perspetiva dos cidadãos e empresas e a necessidade de garantir algumas características da Administração Pública como a versatilidade e a competitividade coloca alguns constrangimentos na conceção das arquiteturas de integração de serviços. Para

que seja possível integrar serviços de forma a que se garanta a mutabilidade da Administração Pública é necessário criar dinamicamente *workflows* que integrem a participação de diversos prestadores de serviços. No entanto, a criação dinâmica de *workflows* suscita algumas preocupações ao nível da segurança motivadas pelo facto de que os participantes no *workflow* podem não ser conhecidos antecipadamente. Daqui surgem as seguintes questões:

- Confiança nos participantes - um participante está acreditado para fornecer o serviço que publicou? É capaz de selecionar o próximo participante a participar na prestação do serviço?
- Preservação da privacidade - parte dos dados manipulados ou produzidos ao longo de um *workflow* poderá ser sensível. Como garantir que os participantes não acedem às partes desses dados de que não são destinatários?
- Conflitos de interesse - um participante poderá recusar-se a participar na prestação do serviço caso exista outro participante com o qual tenha conflitos de interesse?

Foi com base neste levantamento que foram definidas as regras de segurança para a arquitetura que propomos.

5.1.2 Arquitetura e modelo de segurança

A arquitetura baseia-se na troca de mensagens padrão entre Entidades, as quais são os executores dos *workflows*. Permite a criação e execução de workflows dinâmicos porque cada Entidade pode decidir autonomamente usar os serviços de outras Entidades para executar uma parte do *workflow*.

A arquitetura é por desenho versátil, uma vez que permite a adição/remoção de serviços através da publicação e remoção destes mesmos serviços nos repositórios de serviços. Para além disso, as entidades são adicionadas à arquitetura através da publicação de pelo menos um serviço acreditado ou são removidas como consequência da remoção de todos os seus serviços acreditados dos repositórios de serviços.

A arquitetura tem um conjunto de serviços de suporte - Serviço de carimbo temporal, Serviço de Certificação, e Serviço de Diretoria - que permitem a publicação, descoberta e atualização de serviços e a identificação das entidades que estão aptas a colaborar para a prestação de um serviço composto no âmbito de um *workflow* envolvendo a colaboração de várias Entidades. Os serviços prestados pelas Entidades estão associados a um conjunto de estruturas de suporte (*Certificates* e *Service Description*), cujas instâncias contêm a informação necessária ao funcionamento da arquitetura.

Este modelo de segurança suporta a identificação, autenticação, acreditação e autorização, garante que os resultados produzidos pelas entidades apenas são entregues aos seus destinatários, mesmo que estes destinatários não sejam conhecidos na altura da produção do resultado. O modelo também permite que as entidades possam decidir sobre a sua participação e sobre a participação de outras entidades. Estas propriedades são asseguradas através da acreditação de pares entidade-serviço com *Certificates*, os quais possuem os elementos necessários, e de forma confiável, à implantação do modelo de segurança.

Cada organismo da Administração Pública ou entidade privada que deseje publicar um serviço na arquitetura pode fazê-lo. A utilização do serviço está dependente da certificação do par {serviço, entidade}. Como demonstrámos durante a descrição, as entidades tomam as

suas decisões independentemente das entidades que prestam o serviço, uma vez que apenas estão dependentes do *workflow* que deve ser seguido. A escolha das entidades que devem dar seguimento à prestação do serviço está apenas dependente dos critérios definidos pela entidade ou do requerente do serviço complexo que deu origem ao *workflow* (cidadão, organismo, etc.). Isto permite a criação de diferentes caminhos durante a prestação de dois serviços similares.

5.1.3 Casos de uso e protótipo

Para validar a arquitetura proposta foram identificados vários casos de uso relacionados com serviços complexos, envolvendo vários prestadores de serviços, que exemplificam a necessidade de uso de *workflows* dinâmicos e esses casos de uso foram concretizados num protótipo desenvolvido para o efeito. O protótipo foi implementado utilizando a plataforma de agentes de software JADE. O mesmo foi desenvolvido com recurso à linguagem Java, implementa as estruturas de dados e simula os serviços prestados pelos organismos da AP. Este protótipo recorre ao WADE, um plug-in do JADE que permite a definição de *workflows* tornando possível a definição do comportamento interno das Entidades.

A implementação efetuada contempla a arquitetura parcialmente. Os serviços de suporte à arquitetura, as funções criptográficas e os módulos referentes à definição, geração e gestão de *workflow* não estão implementados. Isto significa que as capacidades para cifra de mensagens, assinatura digital e geração e gestão de *workflows* dinâmicos não foram implementadas no protótipo porque não eram fundamentais para validar a arquitetura. No entanto, as tecnologias a utilizar nestes casos estão bem documentadas pelo que a sua utilização no protótipo não acrescenta inovação.

Os casos de uso escolhidos para a validação foram os seguintes: consulta médica; candidatura ao ensino superior; processo de permuta de imóvel; e, viagem ao estrangeiro. Estes casos de uso foram escolhidos devido às suas características, permitindo a exemplificação da aplicação da arquitetura em diferentes contextos. Todos os casos de uso foram adaptados para permitir a integração de serviços prestados por várias organizações (ou organismos da Administração Pública), pelo que não refletem a forma como os serviços são prestados atualmente. Essa experimentação permitiu concluir que a arquitetura está adequada para prestar esses serviços usando workflows dinâmicos e que na execução desses workflows os executores dispõem dos mecanismos de segurança adequados para controlar a sua participação, a participação de terceiros e a privacidade dos resultados produzidos no âmbito dos mesmos.

5.2 Trabalho futuro

Foi nosso principal objetivo, com este trabalho, propor uma arquitetura de integração para governo eletrónico que permita a construção de *workflows* dinâmicos e que aborde as questões de segurança relacionadas com a construção de *workflows* dinâmicos. As características que a arquitetura apresenta permitem que seja aplicada a cenários inter-organizacionais, bem como a cenários intra-organizacionais. No entanto, a sua utilização ainda deve ser precedida de algumas melhorias, a diversos níveis.

A otimização das mensagens será um dos pontos a melhorar. Como observámos, é possível reduzir o tamanho das mensagens utilizando, por exemplo, repositórios de certificados, para evitar que estes circulem com as mensagens, e criar um sistema de *cache* temporário para certificados nos agentes, permitindo que exista um menor tráfego de rede quando é necessário aceder aos certificados.

Outro aspeto a melhorar está relacionado com o carácter distribuído da arquitetura. Existem alguns constrangimentos no acesso ao historial da prestação de um serviço complexo no âmbito de um *workflow* dinâmico. Estes constrangimentos afetam do mesmo modo o seguir da informação privada, caso o dono da informação assim o entenda. Tendo estes pontos em consideração, uma melhoria ao nível do acesso ao registo global poderá ser relevante.

O protótipo apresentado tinha diversas condicionantes: os serviços de suporte à arquitetura não foram implementados, as estruturas de suporte, embora totalmente implementadas, foram utilizadas apenas em parte. É necessário completar os serviços em falta para a utilização do protótipo em casos reais.

Uma das características da arquitetura é não estar dependente da forma como os *workflows* são implementados. Isto permite o desenvolvimento de diversas soluções ao nível da implementação das entidades. Sendo assim, um ponto que poderá ser interessante abordar e estudar é o desenho e desenvolvimento de diferentes soluções para a implementação dos *workflows* prestados individualmente por cada Entidade e a sua utilização em ambientes reais.

Outro ponto que poderá ser interessante estudar é a aplicação das propostas apresentadas nesta tese a outras áreas que não o governo eletrónico. Alargar o contexto da arquitetura e analisar a sua aplicabilidade a outras áreas.

Finalmente, as tecnologias estão em constante evolução, os benefícios atuais de uma tecnologia poderão ser ultrapassados pelo surgimento de outra. Sendo assim, é relevante estudar o impacto da implementação da arquitetura com recurso a outras tecnologias.

Apêndice A

Lista de siglas e acrónimos

ABAC	- *Attribute-based Access Control*
AMA	- Agência para a Modernização Administrativa
AP	- Administração Pública
C	- Clínica
CDI	- *Constrained Data Item*
CRPP	- Conservatória do Registo Predial de Portalegre
DAC	- *Discritionary Access Control*
DGAE	- Direção Geral da Administração Escolar
DREC	- Direção Regional de Educação do Centro
DREN	- Direção Regional de Educação do Norte
DTD	- *Document Type Definition*
EC	- Entidade Certificadora
eGIF	- *electronic Government Interoperability Framework*
EIF	- *European Interoperability Framework*
ESTGA	- Escola Superior de Tecnologia e Gestão de Águeda
FEAF	- *Federal Enterprise Architecture Framework*
FIPA	- *Foundation for Intelligent, Physical Agents*
FSC	- Framework de Serviços Comuns
FSSA	- *Family and Social Services Administration*
GIE	- Guia Integração Electrónica
GovML	- *Governmental Markup Language*

G2B - *Government to Business*

G2C - *Government to Citizen*

G2E - *Government to Employees*

G2G - *Government to Government*

H - Hospital

HTTP - *HyperText Transfer Protocol*

IDA - *Interchange of Data between Administration*

IDABC - *Interoperable Delivery of European eGovernment Services to public Administrations, Businesses and Citizens*

ISA - *Interoperability Solutions for Public Administrations*

IVP - *Integrity Verification Procedures*

JADE - *Java Agent DEvelopment Framework*

LC - Laboratório Clínico

MEC - Ministério da Educação e Ciência

MOM - *Message Oriented Middleware*

MP - Município de Portalegre

MS - Ministério da Saúde

OSCI - *Online Services Computer Interface*

PDA - *Personal Digital Assistant*

PI - Plataforma de Integração

PKI - *Public Key Infrastructure*

RBAC - *Role Based Access Control*

SAGA - *Standards and Architectures for e-Government Applications*

SAML - *Security Assertion Markup Language*

SeCO - *Secure Electronic Contracts*

SFP - Serviço de Finanças de Portalegre

SIL - Sistema de Informação Local

SOA - *Service Oriented Architecture*

SOAP - *Simple Object Access Protocol*

SSL - *Secure Socket Layer*

TIC - Tecnologias da Informação e da Comunicação

TLS - *Transport Layer Security*

TP - *Transformation Procedures*

UDDI - *Universal Description Discovery and Integration*

UDI - *Unconstrained Data Item*

UE - União Europeia

UMIC - Agência para a Sociedade do Conhecimento

WADE - *Workflows and Agent Development Environment*

WFT - *well-formed transaction*

WSA - *Web Services Addressing*

WSDL - *Web Service Description Language*

WSIG - *Web Service Integration Gateway*

WSML - *Web Service Modeling Language*

WSMO - *Web Service Modeling Ontology*

WSMS - *Web Services Management System*

WSMX - *Web Service Modelling Execution Environment*

WSRM - *Web Services Reliable Messaging*

WSS - *Web Services Security*

XACML - *eXtensible Access Control Markup Language*

XKMS - *XML Key Management Specification*

XML - *Extensible Markup Language*

Bibliografia

[1] Agência para a Modernização Administrativa. *Interoperabilidade na Administração Pública*. Disponível em `http://www.iap.gov.pt/Default.aspx`, acedido em 11 de Dezembro de 2011.

[2] Agência para a Modernização Administrativa. *Guia integração electrónica*, 2009. Disponível em `http://www.sg.min-saude.pt/NR/rdonlyres/D6EFBE39-22C3-48FA-9052-93941364C77C/17501/GuiaIntegracao_v091.pdf`.

[3] Agência para a Modernização Administrativa. *Framework de Serviços Comuns.* Disponível em `http://www.ama.pt/index.php`, acedido a 7 de Maio de 2008.

[4] UMIC - Agência para a Sociedade do Conhecimento. *Plataforma de interoperabilidade*, 2007. Disponível em `http://www.umic.pt/index.php?option=com_content&task=view&id=2687&Itemid=112`, acedido a 25 de Julho de 2008.

[5] F. Arcieri et al. A layered IT infrastructure for secure interoperability in personal data registry digital government services. In *Proceedings of the 14th International Workshop on Research Issues on Data Engineering: Web Services for E-Commerce and E-Government Applications (RIDE'04)*, páginas 95-102. IEEE Computer Society, 2004.

[6] F. Arcieri et al. Experiences and issues in the realization of e-government services. In *Proceedings Twelfth International Workshop on Research Issues in Data Engineering: Engineering E-Commerce/E-Business Systems (RIDE'02)*, IEEE Computer Society, págs 143-148, 2002.

[7] U. Bartels e F. Steimke. How to Modernize the People Registration Process. In R. Traunmüller, editor, *Proceedings of the Third International Conference on Electronic Government (EGOV'04)*, volume 3183 de *Lecture Notes in Computer Sciende (LNCS)*, páginas 246-249, Springer, 2004.

[8] F. Bellifemine et al. *Developing multi-agent systems with JADE.* John Wiley & Sons, 2007.

[9] A. Bouguettaya et al. WebDG - a platform for e-Government web services. *Conceptual Model for Advanced Application Domains*, páginas 553-565, 2004.

[10] M. Brahim et al. Composing web services on the semantic web. *The VLDB Journal*, 12(4):333-351, Novembro de 2003.

[11] D. Brewer e M. Nash. In *Proceedings of IEEE Symposium on Security and Privacy*, pp. 206-214, 1989.

[12] Cabinet Office - E-Government Unit, *e-Government interoperability framework, version 6.1, 2005*. Disponível em `http://www.cabinetoffice.gov.uk/media/253452/eGIFv6_1(1).pdf`, acedido a 14 de Maio de 2008.

[13] M. Caloyannides et al. US e-government authentication framework and programs. *IT professional*, 5(3):16-21, 2003.

[14] J. Cardoso. *Semantic Web Services: theory, tools, and applications*. IGI Global, 2007.

[15] F. Casati e M. Shan. Dynamic and adaptive composition of e-services. *Information Systems*, 26, 143-163, 2001.

[16] The Chief Information Office Council. *Federal enterprise architecture framework, version 1.1, 1999*. Disponível em `http://www.cio.gov/Documents/fedarch1.pdf`, acedido a 9 de Maio de 2008.

[17] A comparison of commercial and military computer security policies. In *Proceedings of the 1987 IEEE Symposium on Research in Security and Privacy (SP'87)*, IEEE Press, pp. 184-193, 1987.

[18] Comissão Europeia. *European interoperability framework for pan-european eGovernment services, version 1.0*, 2004. Disponível em `http://ec.europa.eu/idabc/servlets/Docd552.pdf?id=19529`, acedido a 10 de Maio de 2008.

[19] E. Curry. Message-oriented middleware. *Middleware for communication* John Wiley & Sons, 2004.

[20] G. Dias. *Arquitectura de Suporte à Integração de Serviços no Governo Electrónico*. Tese de doutoramento, Universidade de Aveiro, 2006.

[21] G. Dias e J. Rafael. A simple model and a distributed architecture for realizing one-stop e-government. *Electronic Commerce Research and Applications*, 6(1): 81-90, 2007.

[22] S. Durbeck, R. Schillinger, e J. Kolter. Security Requirements for a Semantic Service-oriented Architecture. *The Second International Conference on Availability, Reliability and Security (ARES 2007).*, páginas 366-373, IEEE, 2007.

[23] The European Communities. *Linking up Europe: The Importance of Interoperability for eGovernment Services*. Commission staff working paper, 2003. Disponível em `http://ec.europa.eu/idabc/servlets/Doc2bb8.pdf?id=1675`.

[24] Federal Ministry of the Interior, Germany. *Standards and architectures for eGovernment applications, version 4.0*, 2008. Disponível em `http://www.cio.bund.de/SharedDocs/Publikationen/DE/Architekturen-und-Standards/SAGA/saga_4_0_englisch_download.pdf?_blob=publicationFile`, acedido a 21 de Fevereiro de 2011.

[25] D. Ferraiolo e D. Kuhn. Role-Based Access Controls. In *15th National Computer Security Conference*, pp. 554-563, 1992.

[26] R. Fielding et al. Hypertext Transfer Protocol - HTTP/1.1. RFC 2616. Internet Society, Junho 1999. Disponível em `http://www.ietf.org/rfc/rfc2616.txt`, acedido em 10 de Setembro de 2009.

[27] Gartner. *IT Glossary*. `http://www.gartner.com/it-glossary/`, 2012. Acedido a 7 de Dezembro de 2012.

[28] M. Genesereth e S. Ketchpel. Software agents. *Communications of the ACM*, 37(7):48-60, 1994.

[29] M. Greunz, B. Schopp, e J. Haes. Integrating e-government infrastructures through secure XML document containers. In *Proceedings of the 34th Annual Hawaii International Conference on System Sciences (HICSS-34)*, volume 5; páginas 1-10, 2001.

[30] M. Greunz, B. Schopp, e K. Stanoevska-Slabeva Supporting market transaction through XML contracting containers. In *Proceedings of the Sixth Americas Conference on Information Systems (AMCIS'00)*. 2000.

[31] N. K. Hanna. *Transforming Government and Building the Information Society: Challenges and Opportunities for the Developing World*. Springer, 2010.

[32] D. Holmes. *E-gov: e-business strategies for government*. Nicholas Brealey Publishing, 2001.

[33] IEEE Foundation for Intelligent Physical Agents. *Foundation for Intelligent Physical Agents*. Sítio web: `http://www.fipa.org`.

[34] J. Joshi et al. Digital government security infrastructure design challenges. *Computer* 34(2):66-72, 2001.

[35] A. Kaliontzoglou et al. A secure e-Government platform architecture for small to medium sized public organizations. *Electronic Commerce Research and Applications*, 4(2):174-186, 2005.

[36] G. Kavadias e E. Tambouris. GovML: a markup language for describing public services and life events. In M. Wimmer, editor, *Proceedings of the 5th IFIP International Working Conference on Knowledge Management in Electronic Government (KMGov'03)*, volume 2645 de *Lecture Notes in Computer Science (LNCS)*, páginas 106-115, Springer, 2003.

[37] R. Klischewski. Information integration or process integration? How to achieve interoperability in administration. In R. Traunmuller, editor, *Proceedings of the Third International Conference on Electronic Government (EGOV'04)*, volume 3183 de *Lecture Notes in Computer Science (LNCS)*, páginas 57-65. Springer, 2004.

[38] J. Kolter, R. Schillinger, e G. Pernul. Building a Distributed Semantic-aware Security Architecture. *New Approaches for Security, Privacy and Trust in Complex Environments*, páginas 397-408, 2007.

[39] J. Kolter et al. An architecture integrating semantic e-government services. In *Porceedings of the 5th International e-Government Conference (EGOV'06)*, páginas 1-8, Trauner Druck, 2006.

[40] G. Laskaridis, K. Markellos e P. Markellou. E-government and Interoperability Issues. *International Journal of Computer Science and Network Security*, 7(9): 28-38, 2007.

[41] M. Luck, R. Ashri, e M. D'Inverno. *Agent-based Software Development.* Artech House Publishers, 2004.

[42] F. Marques, G. Dias, e A. Zúquete. A general interoperability architecture for e-Government based on agents and Web Services. In *6ª Conferência Ibérica de Sistemas e Tecnologias de Informação*, páginas 338-343, 2011.

[43] F. Marques, G. Dias, e A. Zúquete. Security concerns in e-Government agent-based interoperability. In *Proceedings of ongoing research, general development issues and projects of EGOV 09, 8th International Conference*, páginas 197-204, 2009.

[44] B. Medjahed et al. Infrastructure for e-government web services. *IEEE Internet Computing*, 7(1), 58-65, Janeiro/Fevereiro de 2003.

[45] T. Narciso. *Interoperabilidade Organizacional na Administração Pública.* Dissertação de mestrado, Universidade de Aveiro, 2010.

[46] V. Ndou. E-government for developing countries: opportunities and challenges. *The Electronic Journal on Information Systems in Developing Countries*, 18(1): 1?24, 2004.

[47] Organization for the Advancement of Structured Information Standards (OASIS). *Web Services Security (WSS) TC version 1.1*, OASIS Standard, 28 de Novembro de 2006. Disponível em `https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss`, acedido a 5 de Dezembro de 2008.

[48] Organization for the Advancement of Structured Information Standards (OASIS). *Authentication context for the OASIS Security Assertion Markup Language (SAML)*, V2.0. Norma OASIS, 15 de Março de 2005. Disponível em `http://docs.oasis-open.org/security/saml/v2.0/saml-authn-context-2.0-os.pdf`, acedido a 14 de Janeiro de 2010.

[49] Organization for the Advancement of Structured Information Standards (OASIS). *Assertions and Protocol for the OASIS Security Assertion Markup Language (SAML)*, V2.0. Norma OASIS, 15 de Março de 2005. Disponível em `http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf`, acedido a 14 de Janeiro de 2010.

[50] Organization for the Advancement of Structured Information Standards (OASIS). *Web Services Reliable Messaging (WSRM) version 1.1*, OASIS Standard, 15 de Novembro de 2004. Disponível em `https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsrm`, acedido a 5 de Dezembro de 2008.

[51] OSCI Leitstelle. *OSCI Transport 1.2 - Design Principles, Security Objectives and Mechanisms*, Maio de 2002. Disponível em `http://www.osci.de/materialien/requirements.pdf`, acedido a 21 de Fevereiro de 2008.

[52] OSCI Steering Office. *OSCI-Transport, Version 2 - Web Services Profiling and Extensions Specification*, Abril de 2010. Disponível em `http://www.osci.eu/transport/osci20/20100427/OSCI20_WS-ProfilingAndExtensionSpecification_Edition3.pdf`, acedido a 17 de Maio de 2011.

[53] OSCI Steering Office. *OSCI-Transport, Version 2 - Technical Features Overview*, Junho de 2009. Disponível em `http://www.osci.eu/transport/osci20/20090601/OSCI2_TechnicalFeaturesOverview_EN.pdf`, acedido a 17 de Maio de 2011.

[54] C. Pfleeger e S. Pfleeger. *Security on Computing*, 4ª edição. Prentice Hall, Pearson Education, 2008.

[55] Research Federal Ministry of Education, Science and Technology. *Information and Communication Services Act*. 1997.

[56] O. Rishi et al. Service Oriented Architecture for Business Dynamics: an Agent-based Approach. Em *Emerging Technologies in E-Government*, 19-28, 2008.

[57] L.M.A. Sabucedo et al. Knowledge-based Platform for eGovernment Agents: A Web-based Solution using Semantic Technologies. *Expert Systems with Applications*, 37, 3647-3656, 2010.

[58] B. Schmid. Elektronische Märkte - Merkmale, Organisation und Potentiale. In A. Hermanns & M. Sauter (Eds.), *Handbuch Electronic Commerce*, Vahlen Verlag, 1999.

[59] B. Schopp. Secure Electronic Contracts. In *Swiss Priority Programme for Information and Communication Structures*, páginas 136-138, 2000.

[60] D. Soares. *Interoperabilidade entre sistemas de informação na administração pública*. Tese de doutoramento, Universidade do Minho, 2009.

[61] F. Steimke e M. Hagen. OSCI: A common communications standard for e-government. In R. Traunmüller, editor, *Proceedings of the Second International Conference on Electronic Government (EGOV'03)*, volume 2739 de *Lecture Notes in Computer Science (LNCS)*, páginas 250-255, Springer, 2003.

[62] E. Tambouris. An integrated platform for realising online one-stop government: the eGOV project. In *Proceedings of the 12th International Workshop on Database and Expert Systems Applications (DEXA'01)*, páginas 359-363. IEEE Computer Society, 2001

[63] Telecom Italia SpA. *Java Agent DEvelopment Framework*. Sítio web: `http://jade.tilab.com/`.

[64] Telecom Italia SpA. *Workflow and Agents Development Environment*. Sítio web: `http://jade.tilab.com/wade/index.html`.

[65] United States Department of Defence. *Trusted Computer System Evaluation Criteria*. DoD Standard 5200.28-STD, 1985.

[66] A. Varghese. Preface. Em C. Codagnone e M. Wimmer (Eds) *Roadmapping eGovernment Research: Visions and Measures towards Innovative Governments in 2020*. MY Print snc di Guerinoni Marco & C., 2007.

[67] M. Vintar, M. Kunstelj e A. Leben. Delivering better quality public services through life-event portals. Em *Proc. of the Tenth NISPAcee Annual Conference: Delivering Public Services in CEE Countries: Trends and Developments*. 25-27 de Abril de 2002. Disponível em `http://unpan1.un.org/intradoc/groups/public/documents/nispacee/unpan004382.pdf`

[68] M. Wimmer. eGovernment as a multidisciplinary research field. Em C. Codagnone e M. Wimmer (Eds) *Roadmapping eGovernment Research: Visions and Measures towards Innovative Governments in 2020*. MY Print snc di Guerinoni Marco & C., 2007.

[69] M. Wimmer e C. Codagnone. Definitions for eGovernment. Em *Roadmapping eGovernment Research: Visions and Measures towards Innovative Governments in 2020*. MY Print snc di Guerinoni Marco & C., 2007.

[70] M. Wimmer. A European perspective towards online one-stop government: the eGOV project. *Electronic Commerce Research and Applications*, 1(1):92-103, 2002.

[71] World Wide Web Consortium (W3C). *Extensible Markup Language (XML) 1.0*, quinta edição, W3C Recommendation, 26 de Novembro de 2008. Disponível em `http://www.w3.org/TR/2008/REC-xml-20081126/`, acedido a 5 de Dezembro de 2009.

[72] World Wide Web Consortium (W3C). *XML Signature Syntax and Processing*, segunda edição, W3C Recommendation, 10 de Junho de 2008. Disponível em `http://www.w3.org/TR/xmldsig-core/`, acedido a 12 de Setembro de 2008.

[73] World Wide Web Consortium (W3C). *SOAP 1.2 Part 1: Messaging Framework*, segunda edição, W3C Recommendation, 27 de Abril de 2007. Disponível em `http://www.w3.org/TR/soap12-part1/`, acedido a 12 de Setembro de 2008.

[74] World Wide Web Consortium (W3C). *Web Service Addressing (WS-Addressing)*, W3C Member Submission, 10 de Agosto de 2004. Disponível em `http://www.w3.org/Submission/ws-addressing/`, acedido a 5 de Dezembro de 2008.

[75] World Wide Web Consortium (W3C). *Web Services Architecture*, W3C Working Group Note, 11 de Fevereiro de 2004. Disponível em `http://www.w3.org/TR/ws-arch`, acedido a 13 de Janeiro de 2009.

[76] World Wide Web Consortium (W3C). *XML Encryption Syntax and Processing*, W3C Recommendation, 10 de Dezembro de 2002. Disponível em urlhttp://www.w3.org/TR/xmlenc-core/, acedido a 12 de Setembro de 2008.