Departamento de Ciências e Tecnologia da Informação

# Controlo e monitorização de informação clínica partilhada

Fábio André Rodrigues Martins

Dissertação submetida como requisito parcial para obtenção do grau de

Mestre em Engenharia Informática

Orientador(a):
Doutor Carlos José Corredoura Serrão, Professor Auxiliar,
ISCTE-IUL

Co-orientador(a):
Mestre Ricardo Alexandre Schiller Wolf Alegre Pinto,
Agap2IT

Setembro, 2016

## Agradecimentos

Quero agradecer aos meus pais, que me proporcionaram a oportunidade de continuar a estudar, assim como, ao meu irmão, Ruben, pelo apoio nesta etapa da minha vida e por último um agradecimento especial à minha namorada, Cláudia, que esteve presente nos bons e maus momentos. Quero agradecer também ao meu orientador, Carlos Serrão, e co-orientador, Ricardo Pinto, pela ajuda e orientação dada na realização desta dissertação.

Por fim, agradeço aos amigos e professores que me acompanharam e ajudaram ao longo do meu percurso académico até chegar a este momento.

## Resumo

Na atual era da informação, sistemas informáticos partilham informação entre si através mecanismos como, por exemplo, as redes de computadores. A informação que é partilhada por esses sistemas informáticos poderá ser classificada de várias maneiras. Uma dessas maneiras baseia-se na sua sensibilidade.

Um exemplo de informação sensível consiste na informação clínica. A razão tem a ver com o facto de esta poder ser utilizada para descriminar, assediar, extorquir ou até condicionar a vida das pessoas. Dada essa possibilidade, é imperativo que haja cautela por parte dos detentores da informação. Essa cautela poderá refletir-se sobre com quem a informação é partilhada.

Apesar da cautela, o detentor da informação poderá, por erro ou ludíbrio, partilhar a informação clínica com alguém menos fidedigno. Este poderá utilizá-la para prejudicar, de algum modo, as pessoas a quem a informação refere. Por essa razão, a seguinte dissertação apresenta uma solução que tem como objetivo permitir ao detentor da informação controlar e monitorizar o que está a ser feito com a informação clínica que este partilhou.

A solução tem como base a utilização de tecnologias de *Information Right Management*. Essas tecnologias são utilizadas para controlar e monitorizar documentos sensíveis partilhados. Nesta dissertação, essas tecnologias serão adaptadas de modo a poder controlar e monitorizar informação partilhada através de outros mecanismos que não documentos.



**Palavras-Chave:** Informação Clínica, Controlo e Monitorização, Segurança da Informação, *Web Services*, *Fast Healthcare Interoperability Resources*, *Information Rights Management*

## Abstract

In the current information age, computer systems share information with each other through computer networks. The shared information can be classified in various ways. One of those ways is its sensitivity.

One example of sensitive information is the clinical information. The main reason has to do with the fact that it can be used to discriminate, harass, extort or to condition the lives of a person. Given this possibility, it is imperative that there is caution on the part of information holders. This caution may be reflected on with whom the information is shared.

Despite the caution, the holder of the information may be mistaken to share clinical information with someone less reliable. He may use it to harm the people who shared information relates to. For this reason, the following dissertation presents a solution that aims to allow the holder of the information to control and monitor what is being done with the shared clinical information.

The solution is based on Information Rights Management technologies. These technologies are used to control and monitor shared sensitive documents. In this dissertation, these technologies will be adapted so as to control and monitor information made available through mechanisms other than documents.

**Key-Words:** Clinical information, Control and Monitoring, Information security, Web Services, Fast Healthcare Interoperability Resources, Information Rights Management

# Índice

# Índice de Ilustrações

# Índice de Tabelas

# Índice de Esquemas

# Capítulo 1
# Introdução

Este capítulo tem o objetivo principal explicar a importância para a realização desta dissertação. Para tal, será explicado as preocupações existentes que resultam das limitações que as tecnologias vigentes dispõem. Tendo como base essas limitações, foram definidos um conjunto de objetivos. De modo a explorar esse objetivos, foi explicado a metodologia de investigação utilizada. Por fim, tendo como base a metodologia, será explicado como o documento está organizado.

## 1.1 Motivação

Com o crescente recurso à utilização de registos clínicos eletrónicos, surgem plataformas *online* que permitem ao paciente consultar os seus registos clínicos (FMUP, 2006; Serviços Partilhados do Ministério da Saúde, 2014). Algumas destas plataformas são de domínio público, como por exemplo o Portal do Utente (Serviços Partilhados do Ministério da Saúde, 2014), enquanto outras são do domínio privado, como por exemplo o LinkedCare (LinkedCare, 2016). O que estas e outras plataformas e aplicações deste género têm em comum consiste no acesso a registos clínicos do paciente. Com o possível aparecimento de cada vez mais aplicações que interagem direta ou indiretamente com a informação clínica de pacientes, e dada a sensibilidade dessa informação, torna-se imperativo controlar o que estas plataformas ou aplicações podem fazer ou estão a tentar fazer, com a informação que tiveram acesso.

## 1.2 Enquadramento

A utilização de registos clínicos eletrónicos apresentam várias vantagens tanto para os prestadores de cuidados de saúde como para os pacientes (FMUP, 2006). Para os prestadores de cuidados, as vantagens passam pelo acesso facilitado à informação, maior legibilidade, possibilidade de troca de informação clínica entre cuidados primários e de especialidade, entre outros (FMUP, 2006). Para o paciente, isto poderá significar uma melhoria nos cuidados de saúde recebidos. Para além desta possível melhoria, a utilização de registos clínicos eletrónicos permite que o paciente possa consultar os seus registos através de plataformas *online*, tais como o Portal do Utente[1] (Serviços Partilhados do Ministério da Saúde, 2014) ou o LinkedCare[2], uma versão proprietária (LinkedCare, 2016).

Para poder tirar partido das vantagens na utilização de registos clínicos eletrónicos, são disponibilizados serviços que permitem a interoperabilidade entre os diversos sistemas. Esses serviços poderão ser implementados utilizando mecanismos de interoperabilidade baseados em serviços web RESTful (HL7, 2015).

A utilização de registos clínicos eletrónicos oferece vantagens mas também riscos. Um desses riscos consiste na possibilidade de existirem potenciais quebras de segurança (FMUP, 2006). Um exemplo de quebra de segurança consiste na partilha da informação clínica com uma aplicação ou plataforma menos fidedigna que poderá utilizar essa informação para fins menos corretos, como por exemplo, a divulgação não autorizada. Os pacientes podem ser prejudicados ou ficar debilitados caso exista algum acesso ou divulgação não autorizada da sua informação clínica, no sentido de esta ser utilizada para descriminar, assediar, extorquir ou até condicionar a sua vida (Michael McFarland, 2012).

[1] https://servicos.min-saude.pt/utente/
[2] http://www.linkedcare.com/index.html

Ao partilhar registos clínicos eletrónicos com plataformas, tais como o LinkedCare, existe sempre esse risco. Apesar de ser possível uma ação judicial, o estrago já foi feito e os pacientes já poderão estar a ser prejudicados. Para evitar tal situação é necessário controlar e monitorizar o que aplicações ou plataformas estão a fazer ou a tentar fazer com a informação que tiveram acesso.

A solução poderá partir da ideia por detrás das tecnologias IRM (*Information Rights Management*) (Geoff Anderson, 2008). Estas tecnologias são utilizadas para controlar e monitorizar documentos sensíveis partilhados (Geoff Anderson, 2008). É importante referir que estas tecnologias funcionam com informação partilhada em formato de documento, não com informação partilhada utilizando o método de interoperabilidade baseado em serviços web RESTful (Information Rights Management, 2015). Com base desta limitação das tecnologias IRM, surgem as seguintes questões de investigação:

- Será possível implementar um mecanismo, tendo como base as ideias das tecnologias IRM, que permita controlar e monitorizar informação clínica partilhada via RESTful?
- Quais as limitações que esse mecanismo terá?

## 1.3 Objetivos

O objetivo principal deste trabalho consiste em apresentar uma solução que permitam aos sistemas que disponibilizam acesso a registos clínicos, controlar e monitorizar as operações que as aplicações e plataformas estão a realizar com esses registos.

De modo a atingir o principal objetivo de controlar a utilização de registos clínicos, foram definidos um conjunto de pequenos objetivos que a solução deverá atingir:

- Conseguir partilhar registos clínicos em formatos utilizados em serviços web;
- Implementar ou utilizar um mecanismo que permita que aplicações não-autorizadas não tenham acesso direto aos registos clínicos disponibilizados;
- Permitir controlar as ações que um utilizador pode realizar quando está a aceder a informação clinica, tais como, visualização, impressão, cópia, captura de ecrã, entre outras;
- Conseguir registar todas as ações realizadas ou tentadas sobre os registos clínicos.

## 1.4 Metodologia

A metodologia define-se como a arte de dirigir o espírito na investigação da verdade. Nesta dissertação a arte escolhida designa-se por *Design Science Research* (DSR).

O DSR define-se como o desenho e investigação de artefactos num determinado contexto. Por outras palavras, significa investigar o problema a partir de uma solução que foi desenvolvida para esse efeito. Tendo como base essa ideia, o DRS define um ciclo de etapas para o desenvolvimento da solução. Nesta dissertação, as etapas correspondem às seguintes:

- **Investigação do Problema** - Será analisado o contexto onde o problema se insere;
- **Definição da Solução** - Nesta etapa será definida o comportamento ideal que a solução deverá ter, de modo a resolver o problema. Esta terá em conta os objetivos definidos nesta dissertação;
- **Implementação da Solução** - Será descrito como a solução foi implementada. Esta tem como base o comportamento ideal definido;
- **Demostração -** Utilizando a solução desenvolvida na implementação, será demostrado como esta funciona;
- **Validação** - Tendo como base as considerações definidas nos Objetivos, será analisado se o comportamento verificado poderá satisfazer tais considerações.

## 1.5 Organização do Documento

O presente documento está dividido nos seguintes capítulos: Revisão da Literatura; Análise Desenho e Arquitetura; Implementação; Validação; Conclusão.

Ao longo da Revisão da Literatura ir-se-á apresentar informação sobre a atual representação de registos clínicos em formato eletrónico e quais as suas vantagens. De seguida é apresentada a forma como os registos clínicos em formato eletrónico são representados e partilhados entre os vários sistemas hospitalares e não hospitalares. Isto leva a uma apresentação das limitações existentes aquando da partilha de informação clínica. De modo a fazer frente a essas limitações será apresentado o modo como as tecnologias existentes atuam em situações semelhantes. Nos capítulos subsequentes será detalhado como essas tecnologias poderão ser utilizadas.

No capítulo da Análise Desenho e Arquitetura, será apresentada como uma solução ideal deveria comportar-se para controlar e monitorizar informação clínica partilhada. Esta tem como base o modo como as tecnologias existentes atuam em situações semelhantes, assim como, os objetivos definidos.

De seguida, foi desenvolvida uma prova de conceito. No capítulo da Implementação, é demostrado como a prova de conceito foi implementada. Após a implementação, é demostrado na secção da Demonstração, a solução em funcionamento.

Por último, será apresentado no capítulo da Conclusão, as conclusões sobre as questões de investigação definidas, tendo como base a implementação e demostração, assim como, os objetivos. Para além disso, serão apresentadas possíveis investigações a realizar como continuidade desta dissertação.

# Capítulo 2
# Revisão da Literatura

Com o advento da Internet, as redes de computadores diversificaram-se a nível global. Essas redes permitiram uma comunicação rápida entre sistemas e/ou pessoas, independentemente da sua localização. Essa capacidade tem uma enorme utilidade em diversas áreas. Uma dessas áreas é a saúde.

Na saúde, as redes de computadores permitem que os profissionais de saúde possam aceder a toda a informação clínica do paciente com maior facilidade, incluindo os cuidados recebidos noutras instituições clínicas. Para tal, basta que os profissionais de saúde utilizem um sistema de informação de saúde de modo a que possam registar e partilhar os registos clínicos.

O presente capítulo irá apresentar uma investigação com o intuito de perceber como sistemas de informação de saúde partilham informação entre si. Isto envolve perceber como os sistemas funcionam e como representam a informação. Após a identificação das limitações será apresentado o modo como os sistemas vigentes lidam com problemas semelhantes aos identificados.

## 2.1 Sistemas de Informação de Saúde

Um registo clínico consiste na informação que é registada por um profissional de saúde sobre um paciente que procurou auxílio médico. Estes registos contêm informação como exames, diagnósticos, tratamentos, resultados dos tratamentos entre outros (FMUP, 2006).

Um registo clínico pode ser realizado de duas maneiras - em papel ou em formato eletrónico. O formato eletrónico tem vantagens em comparação com o formato papel, pois permite auxiliar na prestação de cuidados de saúde, na decisão clínica, na investigação, na educação dos prestadores de cuidados de saúde, assim como permitir avaliar os cuidados prestados de modo a alcançar um melhor planeamento e gestão dos recursos de saúde. Este também permite facilmente a utilização e integração de informação clínica de diversas fontes (FMUP, 2006).

Quando um paciente é atendido por um profissional de saúde, este poderá requisitar exames em serviços hospitalares que contêm sistemas de informação próprios. Estes sistemas, para realizarem o exame requisitado, poderão necessitar de informações sobre paciente. Ao utilizar um formato eletrónico de registos clínicos, esta necessidade é facilmente satisfeita, quando comparado com registos em formato papel.

Para além de permitir a troca de informação entre sistemas hospitalares, os registos eletrónicos de informação clínica permitem que o paciente possa consultar os seus registos clínicos com maior facilidade. Isto é feito através de plataformas *online*, tais como o Portal do Utente, criado pelo Mistério da Saúde e que está integrado com a PDS (Plataforma de Dados da Saúde) do SNS (Sistema Nacional de Saúde) em Portugal (Serviços Partilhados do Ministério da Saúde, 2014), ou o LinkedCare, uma versão proprietária do Portal do Utente (LinkedCare, 2016). Estas plataformas, para além permitirem consultar a informação clínica, também permitem marcar consultas, consultar receitas de medicamentos, consultar exames realizados, entre outros (Serviços Partilhados do Ministério da Saúde, 2014).

Para que os sistemas de serviços hospitalares comuniquem e partilhem informação entre si e com as plataformas *online*, é imperativo que compreendam o que está a ser comunicado.

Para compreenderem o que está a ser comunicado, os sistemas hospitalares e plataformas devem acordar na norma de comunicação. Existem várias normas para a troca de informação clinica. Algumas dessas normas são específicas de um domínio, como por exemplo, troca de resultados laboratoriais entre hospitais e laboratórios (California Health Care Foundation, 2014), outras são mais abrangentes, como por exemplo o HL7 (HL7, 2016). Esta ultima é tipicamente utilizada em sistemas hospitalares (Dreyer, 2000) e tem ganho popularidade como uma norma flexível na troca de informação clinica estruturada (HIMSS, 2015). Ao utilizar uma norma abrangente, ao invés de várias normas específicas, poderá significar uma redução do esforço necessário para a implementação.

### 2.1.1 Health Level Seven

A norma HL7 define uma *Framework*, e normas relacionadas, para troca, integração, partilha e requisição de registos clínicos em formato eletrónico (HL7, 2016).

O HL7 encontra-se atualmente na sua versão 3 (Corepoint Health, 2010). Apesar disso, o HL7 versão 2 ainda é bastante utilizado, especialmente em sistemas antigos (Corepoint Health, 2010). A versão 2 do HL7 foi desenvolvida de modo a que cada sistema hospitalar pudesse adaptar a norma à sua medida (Corepoint Health, 2010). Este facto leva a que, ao desenvolver uma aplicação que interage com vários sistemas hospitalares, seja necessário implementar uma interface de comunicação específica para cada um deles.

A versão 3 do HL7 veio tentar resolver a questão, mas a sua adoção é cara e irá demorar bastante tempo (Corepoint Health, 2010). Outro facto que poderá fazer demorar a adoção, consiste na incompatibilidade da versão 3 com a versão 2 (Corepoint Health, 2010). Isto significa que as aplicações que utilizam a versão 3 têm de suportar a versão 2 do HL7, para que possam comunicar com sistemas antigos (Corepoint Health, 2010).

Para o intercâmbio, a norma HL7 apenas define a utilização de protocolos MLLP (*Minimal Lower Layer Protocol*) (HL7, 2016), mais conhecidos como LLP (*Lower Layer Protocol*). O LLP corresponde aos protocolos que são definidos nas camadas abaixo da camada aplicação do modelo TCP/IP[3] (Howe, 1999). Isto significa que o HL7 não define nenhum protocolo específico na camada aplicação. Isto leva a que cada sistema de informação hospitalar implemente o seu próprio protocolo de nível aplicação, conduzindo a uma maior dificuldade na integração.

### 2.1.2 Open Electronic Health Records

A dificuldade identificada no HL7 não é nova. Varias soluções, tais como *Open Electronic Health Records* (OpenEHR[4]) têm surgido como forma de responder a essas mesmas limitações do HL7.

O OpenEHR consiste numa comunidade virtual com o intuito de transformar informação clínica de formato físico para formato digital, assegurando uma interoperabilidade universal (OpenEHR, 2016). Para tal, o OpenEHR define um conjunto de modelos genéricos, denominados de *Reference Model*, que são utilizados para representar qualquer informação clínica no exterior (OpenEHR, 2016). Para chegar a esse modelo, o OpenEHR define um conjunto de camadas que transformam a informação clínica de modo a encaixar no modelo genérico (OpenEHR, 2016).

### 2.1.3 Fast Healthcare Interoperability Resources

Reconhecendo dos desafios da norma HL7, a organização que definiu o HL7 desenvolveu a sua própria norma denominada de FHIR (HL7, 2015). O FHIR consiste numa *Framework* norma com o objetivo de fornecer mecanismos de interoperabilidade para o HL7, baseados nas normas existentes na *web*, tais como XML, JSON[5], HTTP, OAuth, entre outros (HL7, 2015). Este suporta arquiteturas baseadas em RESTful e é suficientemente flexível para ser utilizado em diversos contextos, tais como aplicações *mobile* ou partilha de registos clínicos eletrónicos (HL7, 2015).

[3] O principal modelo de comunicação na Internet. Está dividido em camadas e cada camada contem protocolos que resolvem problemas de rede específicos dessa camada.
[4] http://www.openehr.org/home
[5] Um formato leve de troca de dados, processável por uma máquina, fácil de escrever e ler por um humano. Este é independente de qualquer linguagem.

O FHIR funciona mapeando mensagens HL7 para mensagens num formato que a organização que definiu o FHIR define como uma norma e que são reconhecidos por outros sistemas de informação de saúde (FHIR, 2015). Estas mensagens podem ser expressas tanto em XML com JSON e acedidos via RESTful (FHIR, 2015). No FHIR, os diversos formatos norma são denominados de recursos (FHIR, 2015). Para mapear uma mensagem HL7 para um recurso FHIR, poderão ser utilizadas ferramentas de integração, como por exemplo Iguana[6] (Interfaceware, 2016). Após esse mapeamento, os recursos FHIR estão em condições para serem disponibilizados para o exterior, através de um serviço *Web*.

## 2.2 Serviços Web

Um Serviço *Web*, ou simplesmente *Web Service*, consiste num sistema de *software* que foi desenhado para suportar interações máquina-para-máquina pela rede, através de uma interface de comunicação, preferencialmente, *loosely-coupled*[7] (Merenyi, 2013). Este não está ligado a nenhuma linguagem de programação ou sistema operativo em particular (Alonso, Casati, Kuno, & Machiraju, 2004). Os principais *Web Services* são os baseados em REST e em SOAP (Pingdom, 2010).

### 2.2.1 SOAP

O SOAP (*Simple Object Access Protocol*) consiste num método de interoperabilidade baseado em XML que veio criar um conjunto de normas que simplificaram o modo como a comunicação entre sistemas distribuídos era realizada (Alonso, Casati, Kuno, & Machiraju, 2004; Curbera, Duftler, Khalaf, & Nagy, 2002). Este baseia-se no paradigma orientado a serviços em que o sistema expõe um serviço que pode ser invocado por outro sistema (Alonso, Casati, Kuno, & Machiraju, 2004). Este paradigma é imperativo, isto é, o foco está nas operações a disponibilizar e nas entradas e saídas dessas operações (Zhao & Doshi, 2009).

[6] http://www.interfaceware.com/iguana.html
[7] Consiste numa arquitetura que tem o objetivo reduzir o risco de, ao executar uma alteração numa interface disponibilizada, obrigar a quem utiliza essa interface a realizar alterações

Apensar do SOAP ter simplificado o modo como sistemas os distribuídos comunicam entre si a tendência aponta para uma maior preferência no tipo de *Web Services* baseado em REST (Pingdom, 2010). Isto poderá ser resultado do facto de o REST ser mais simples de utilizar e desenvolver, assim como, possibilitar que aplicações *mobile* obtenham uma melhor performance na execução de pedidos ao servidor (Hunsaker, 2015; Steven Davelaar, 2015; Alex Rodriguez, 2015). Os *Web Services* baseados em REST são conhecidos como RESTful (Çetinkaya, 2014).

### 2.2.2 RESTful

O REST (*Representational State Transfer*) define um conjunto de princípios arquitetónicos através dos quais permite desenhar um *Web Service* (Alex Rodriguez, 2015). Este tem ganho uma grande aceitação como uma alternativa a métodos de interoperabilidade mais tradicionais, devido à sua simplicidade de utilização (Alex Rodriguez, 2015).

Um dos princípios do REST consiste no paradigma orientado ao recurso (Zhao & Doshi, 2009). Isto significa que o foco do REST está na descrição e decisão de quais os recursos que devem ser expostos ao exterior (Zhao & Doshi, 2009). Para tal, o REST utiliza as características do protocolo HTTP (*Hypertext Transfer Protocol*) (Pautasso, Zimmermann, & Leymann, 2008). Ao utilizar este protocolo, o REST disponibiliza uma interface de comunicação uniforme (Pautasso, Zimmermann, & Leymann, 2008).

Para identificar um recurso disponibilizado, o protocolo HTTP utiliza URI (*Uniform Resource Identifier*) (Fredrich, 2012). No REST, o URI deve ser auto descritivo, isto é, deve ser claro de modo a ser possível identificar qual o recurso que está a referenciar (Fredrich, 2012). Isto é obtido através de uma organização hierárquica, semelhante a uma estrutura de diretorias, dos recursos identificados no URI (Fredrich, 2012). Estes URI's são denominados de *endpoints*.

Para manipular o recurso identificado no URI, o REST faz uso dos métodos protocolo HTTP: POST, GET, PUT e DELETE (Fredrich, 2012). Estas operações são semelhantes as operações CRUD (*Create, Read, Update*, e *Delete*) em base de dados respetivamente (Fredrich, 2012).

Um outro princípio do REST consiste em ser *stateless*[8] (Fredrich, 2012). Isto implica que sempre que o cliente envia um pedido ao servidor, a mensagem deve conter toda a informação necessário para a sua execução, independentemente de outras mensagens enviadas anteriormente (Fredrich, 2012). Isto permite que o REST tenha um elevado desempenho, sendo por isso, escalável.

É importante que os clientes consigam guardar em *cache* a resposta do servidor, de modo a ter uma maior performance na utilização da largura de banda (Fredrich, 2012). O REST deve suportar uma variedade de formatos de dados na resposta do servidor, tais como XML e JSON (Pautasso, Zimmermann, & Leymann, 2008).

De modo a prevenir acessos indevidos aos recursos disponibilizados é imperativo a utilização de um mecanismo de controlo de acesso.

## 2.3 Controlo de Acesso

Os controlos de acesso fazem parte da segurança numa organização, pois ajudam a proteger os seus ativos (Perrin, 2007). Para tal, um sistema de controlo de acesso têm que garantir a autenticação do sujeito, assim como, garantir que este tem a autorização para executar as ações sobre o objeto (Perrin, 2007).

Um sujeito que seja o titular do objeto, ao usar as suas credenciais, terá um controlo absoluto ou quase absoluto sobre as ações possíveis de realizar sobre esse objeto. Para que uma Aplicação Terceira obtenha acesso aos objetos em nome do titular, de modo a realizar alguma ação ou disponibilizar algum serviço, o sujeito titular teria de partilhar as suas credenciais com a aplicação. Isto iria dar um poder absoluto ou quase absoluto à Aplicação Terceira, podendo ser prejudicial para o titular. Para mitigar esse risco, ao aceder a recursos clínicos via *web*, a norma FHIR recomenda o Oauth1.0 (FHIR Security, 2015).

[8] O servidor não guarda estado sobre os clientes com quem comunica

### 2.3.1 OAuth1.0

O OAuth1.0 consiste num método que permite que aplicações terceiras obtenham acesso a recursos, em nome do proprietário (E. Hammer-Lahav, 2010). Este protocolo dispõe de um sucessor, denominado de OAuth2.0. Apesar disso, o FHIR recomenda o OAuth1.0 (FHIR Security, 2015). Isto poderá dever-se ao facto de que o OAuth1.0, ou simplesmente OAuth, ser considerado mais seguro. A razão de tal afirmação tem a ver com o facto o OAuth2.0 basear-se apenas na utilização SSL/TSL[9] para garantir a obtenção segura de acessos aos recursos (Mayko, 2015). É importante referir que o OAuth2.0 foi desenvolvido com o intuito de simplificar a aquisição de acesso (Mayko, 2015). No OAuth1.0, a autenticação das mensagens é realizada através de assinaturas com recurso a segredos partilhados. (Mayko, 2015). No OAuth2.0, autenticação das mensagens apenas se baseia na utilização do SSL/TSL (Mayko, 2015). Isto simplifica a sua aquisição.

De modo a explicar o OAuth1.0, é necessário definir três atores: o consumidor, que consiste na Aplicação Terceira; o fornecedor, que consiste na aplicação que detém os recursos; e o cliente, que é o proprietário dos recursos.

Para que o consumidor obtenha acesso aos recursos, necessita de trocar um conjunto de mensagens com o fornecedor (E. Hammer-Lahav, 2010). Todas mensagens trocadas poderão utilizar métodos criptográficos, de modo a garantirem a autenticação, integridade e não repudio (E. Hammer-Lahav, 2010).

No OAuth1.0, antes de poder trocar mensagens com o fornecedor, o consumidor necessita de estar registado no fornecedor, de modo a que seja atribuído um indentificador único e uma chave secreta (E. Hammer-Lahav, 2010). Estes serão utilizados para identificar e autenticar o consumidor (E. Hammer-Lahav, 2010).

[9] Uma norma de segurança que cria uma ligação encriptada entre um servidor e um cliente.

Quando um cliente inicia o processo de autorização para que o consumidor aceda aos seus recursos, o consumidor envia um pedido ao fornecedor com o seu indentificador e chave (E. Hammer-Lahav, 2010). Este retorna um *token* temporário que será utilizado para que o consumidor reencaminhe o cliente para o fornecedor, de modo que o cliente confirme que autoriza o acesso aos recursos (E. Hammer-Lahav, 2010). Ao confirmar, o fornecedor redireciona o cliente para o consumidor com um *token* de verificação (E. Hammer-Lahav, 2010). Finalmente, o consumidor utiliza esse *token* de verificação de modo a requisitar ao fornecedor o *token* de acesso final, que permite a este aceder aos recursos em nome do cliente (E. Hammer-Lahav, 2010).

O OAuth1.0 apenas permite controlar quem tem acesso á informação, mas, após a sua divulgação, não permite acompanhar e verificar o que está a ser feito com essa informação. Para tal, é necessário utilizar um mecanismo para controlar e monitorizar informação que foi partilhada.

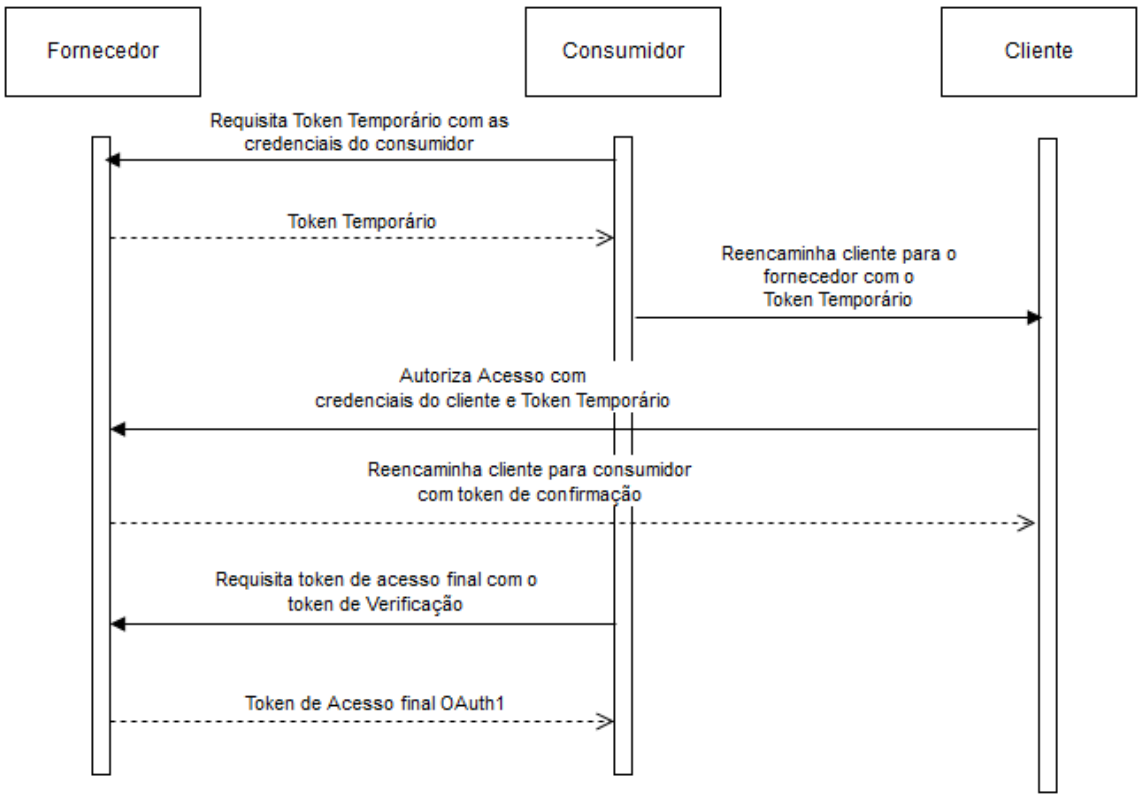



**Ilustração 1 - Interação do protocolo OAuth1.0**

## 2.4 Controlo e Monitorização de Informação Partilhada

Controlar e monitorizar informação partilhada não é algo novo. A ideia inicial tinha o objetivo de proteger conteúdo sujeito a direitos de autor. Nessa altura denominava-se de DRM (*Digital Rights Management*). Este termo consiste no conjunto de tecnologias utilizadas para o controlar o acesso, visualização e distribuição de conteúdo digital sujeito a direitos de autor, de modo a proteger contra duplicação, alteração e distribuição não autorizada (Pal, 2014).

### 2.4.1 Digital Rights Management

As soluções DRM baseiam-se em envolver, ou colocar, o conteúdo digital num contentor, que o protege e define o ciclo de vida, regras de utilização, pagamentos e restrições de distribuição (Pal, 2014).

Uma das principais filosofias do DRM consiste na separação do conteúdo digital dos direitos de utilizar esse conteúdo. Isto permite que o conteúdo seja distribuído livremente, mas o consumo só pode ser realizado se o consumidor tiver uma licença válida. Para que o conteúdo seja distribuído livremente, sem que seja consumido, é necessário utilizar algum tipo de encriptação do conteúdo digital. Neste contexto, a licença ganha grande importância pois, para além de conter as condições de utilização do conteúdo digital, contém uma chave que permite desencriptar o conteúdo digital encriptado (Subramanya & Yi, 2006; Pal, 2014) .

A escolha da tecnologia, e o modo de a implementar, depende do tipo de conteúdo, necessidades da aplicação e tolerância para utilização indevida (Subramanya & Yi, 2006) É de notar que este tipo de tecnologias são utilizadas para proteger conteúdo sujeito a direitos de autor, como por exemplo, conteúdos multimédia. No caso se ser necessário controlar e monitorizar documentação sensível partilhada, é necessário utilizar um subconjunto das tecnologias DRM denominado de *Information Rights Management*.

### 2.4.2 Information Rights Management

O IRM (*Information Rights Management*) consiste nas tecnologias utilizadas para controlar a divulgação e utilização de informação considerada sensível, após a sua partilha (Geoff Anderson, 2008). Estas tecnologias são baseadas no *Digital Rights Management* (DRM), mas, ao contrário do DRM que tenta proteger a propriedade intelectual, o IRM tenta proteger a informação (Geoff Anderson, 2008). É importante reconhecer que o IRM não irá proteger completamente contra a divulgação não autorizada de informação sensível, mas sim reduzir risco (Geoff Anderson, 2008).

O IRM é desenhado para implementar as políticas a nível das aplicações. Isto significa que é a aplicação que é responsável por garantir a aplicação das políticas (Oracle; Geoff Anderson, 2008). Tanto a Oracle como a Microsoft dispõem de aplicações que implementam tecnologias de IRM (Oracle; Geoff Anderson, 2008), como por exemplo o Microsoft Office. Apesar de serem de empresas diferentes, ambas implementam o IRM de modo semelhante. A base dessas implementações consiste na encriptação (Oracle; Geoff Anderson, 2008). Isto garante que apenas aplicações autorizadas podem abrir e ler o documento.

Um documento com informação é encriptado e é atribuído um conjunto de permissões de acesso e de utilização pelo autor. Quando outro utilizador tenta aceder, a aplicação utilizada para ler esse documento, irá utilizar a tecnologia IRM para consultar um ou mais servidores de modo a confirmar as permissões de acesso, utilização, assim como, a identidade do utilizador. Dado a confirmação da identidade, a tecnologia irá dar acesso e forçar o cumprimento das permissões de acesso e de utilização do documento (Oracle; Geoff Anderson, 2008). A tecnologia também monitoriza todas as ações autorizadas e não autorizadas que o utilizador tenta realizar sobre o documento (Oracle; Geoff Anderson, 2008).

O autor do documento poderá revogar as permissões de acesso e utilização do documento sempre que deseja, isto porque, a tecnologia consulta o Servidor de Permissões antes de permitir a realização de qualquer ação (Oracle; Geoff Anderson, 2008). É importante referir que as tecnologias IRM funcionam quando a informação que é partilhada está em formato de documento (Information Rights Management, 2015).

# Capítulo 3

# Análise, Desenho e Arquitetura

A solução apresentada tem o objetivo de controlar e monitorizar a informação clínica partilhada. Para tal foram definidos nos Objetivos, um conjunto de considerações. De modo a tê-las em conta, foi analisado o modo como as tecnologias existentes atuam em situações semelhantes. Com base na análise do capítulo da Revisão da Literatura, conclui-se que estas utilizam uma aplicação para controlar e monitorizar a informação partilhada. A grande limitação dessas tecnologias consiste no facto que a informação partilhada está num formato de documento. Isto significa que as tecnologias não funcionam a não ser que a informação esteja num formato de documento.

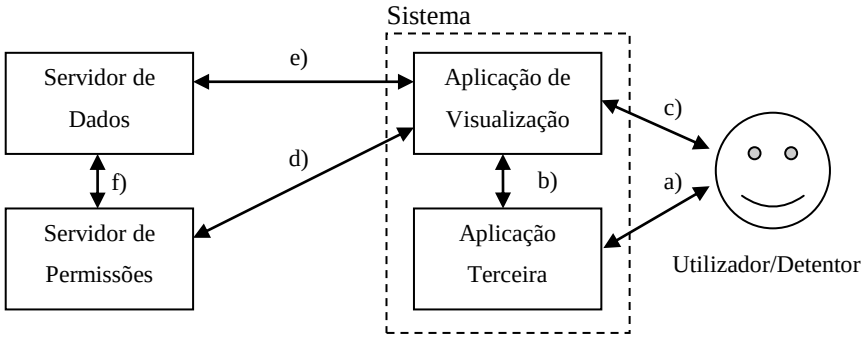
Aproveitando a ideia por detrás das tecnologias para controlar e monitorizar a informação partilhada e adaptado às limitações, define-se os seguintes funcionalidades principais:

- Criação de uma aplicação que irá ser responsável por receber a informação clínica partilhada e apresentá-la ao utilizador. Este deverá seguir as diretrizes indicadas nas tecnologias IRM, com a exceção do formato da informação partilhada. Esta deverá ser no formato partilhado via RESTful:
    - A informação que é partilhada via RESTful, consiste em informação clínica seguindo a norma FHIR;
- A aplicação nunca deverá partilhar a informação recebida com outras aplicações;
- As permissões especificadas no Servidor de Permissões devem ser relativas às permissões de manipulação que o utilizador poderá realizar, tais como, captura de ecrã ou cópia (*copy paste*) entre outros.

Posto isto, o atual capítulo irá apresentar como as funcionalidades definidas serão ser satisfeitas. Para tal foi definido um modelo conceptual de uma arquitetura de sistemas no qual contem os todos os componentes necessário para a solução. Utilizando esse modelo, será descrito como cada componente funciona e interage com os restantes.

## 3.1 Modelo Conceptual

De modo a apresentar a solução para o controlo e monitorização de informação clínica partilhada, será utilizado o seguinte modelo conceptual de uma arquitetura de sistemas.




**Esquema 1 - Modelo conceptual de uma arquitetura de sistemas**

O modelo conceptual apresentado no Esquema 1 mostra um conjunto de elementos que interagem entre si. As interações identificadas assumem que a troca de informação entre os elementos é realizada utilizando canais de comunicação seguros. Os mecanismos que tornam os canais seguros poderão ser diferentes consoante a implementação.

### 3.1.1 Utilizador/Detentor

As interações no modelo conceptual apresentado no Esquema 1 têm início no utilizador/detentor. Este representa o utilizador que é detentor de alguma informação clínica, isto é, dispõe de credenciais com permissões que lhe permitem aceder e/ou manipular informação clínica. Para tal o utilizador/detentor interage com um conjunto de aplicações numa perspetiva de um utilizador final.

**Tabela 1 - Formato das credenciais do Utilizado/Detentor**

| Nome do utilizador | Palavra-chave |
|---|---|

É importante referir que as aplicações que o utilizador/detentor irá interagir para aceder e/ou manipular informação clínica, se encontram em funcionamento no mesmo sistema. Estas aplicações consistem na Aplicação de Visualização e Aplicação Terceira.

### 3.1.2 Aplicação Terceira

A Aplicação Terceira consiste numa aplicação que foi desenvolvida pela entidade que necessita da informação clínica. Esta deve possuir credenciais únicas de modo a poder ser identificada inequivocamente sempre que necessário. A Aplicação Terceira é responsável por garantir a segurança das credenciais que lhe foram atribuídas pela entidade competente. É importante referir que associadas às credenciais da Aplicação Terceira estão permissões de acesso.

**Tabela 2 - Formato das credenciais da Aplicação Terceira**

| Identificador Único | Segredo<br>(corresponde a uma chave) |
|---|---|

A interação com a Aplicação Terceira, identificada como Esquema 1 - a), representa as interações requeridas ao utilizador/detentor, de modo a cumprir regras de negócio da entidade que necessita da informação clínica. Algumas dessas regras de negócio poderão implicar o acesso e/ou manipulação da informação clínica. Para tal a Aplicação Terceira terá de comunicar e interagir com outra aplicação.

A comunicação com outra aplicação é realizada através de IPC's[11]. Estes consistem em mecanismos que permitem que processos independentes comuniquem entre si. Os IPC's a utilizar para a comunicação, vão depender do sistema onde as aplicações então a operar, assim como, quais destes as aplicações estão preparadas para aceitar.

As interações da Aplicação Terceira com outra aplicação têm o objetivo de permitir que a Aplicação Terceira possa delegar ações a outra aplicação. Essas ações podem ser de dois tipos:

- O primeiro tipo de ação, consiste em pedir que essa outra aplicação requisite ao utilizador/detentor, em nome da Aplicação Terceira, a possibilidade de aceder e/ou manipular a informação clínica em nome do utilizador detentor. Isto implica que a Aplicação Terceira partilhe com outra aplicação, as suas credenciais;

[11] *Inter-Process Communication*

- O segundo tipo de ação consiste em pedir à outra aplicação, que mostre a informação clínica. Isto também implica que a Aplicação Terceira partilhe com a outra aplicação, as suas credenciais, de modo a que esta consiga autenticar a sua proveniência.

A aplicação que a Aplicação Terceira irá interagir denomina-se de Aplicação de Visualização. Estas interações poderão ser identificadas no Esquema 1 - b).

O seguinte diagrama ilustra a sequência principal de interação entre o utilizador/detentor, Aplicação terceira e de Visualização. No decorrer da descrição do modelo conceitual, mias detalhes serão acrescentados aos subdiagramas de sequência.

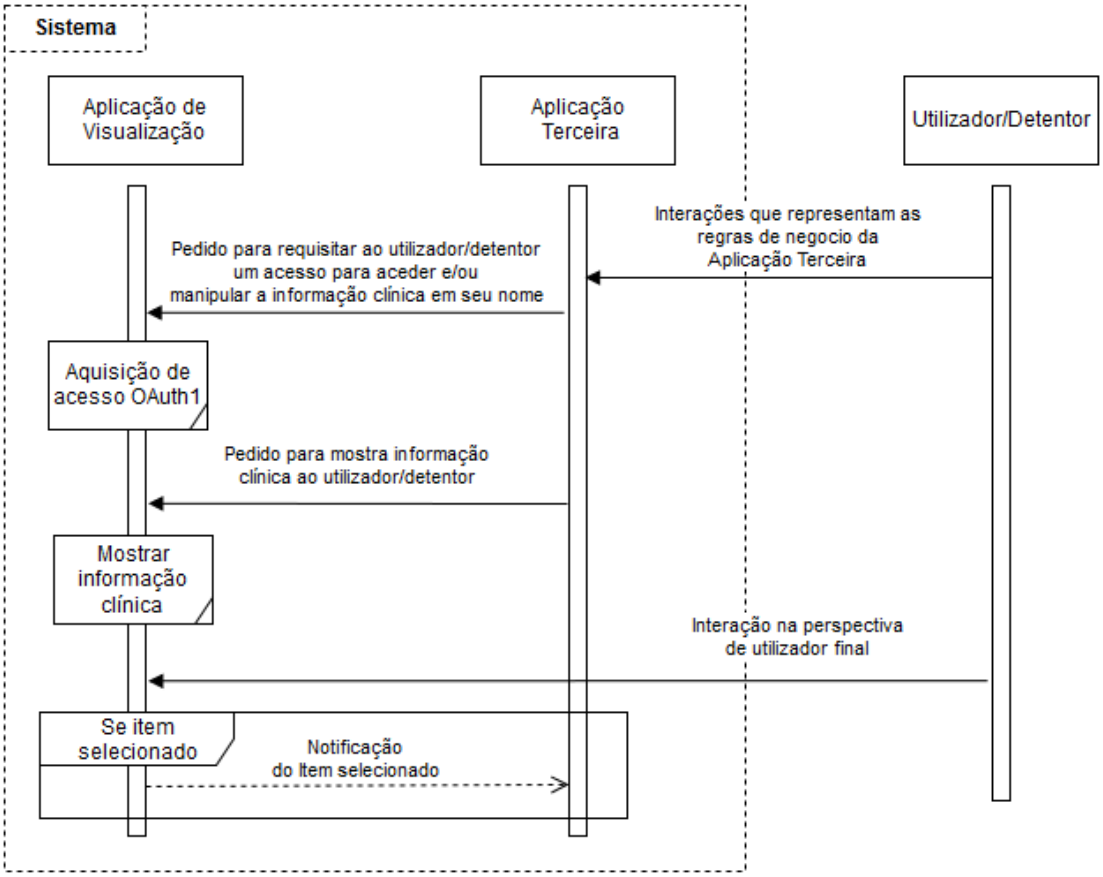



**Ilustração 2 - Diagrama de sequência das interação entre Aplicação de Visualização, Terceira e Utilizador/Detentor**

### 3.1.3 Aplicação de Visualização

A Aplicação de Visualização consiste na aplicação com o objetivo de encapsular e proteger a informação. Esta foi desenvolvida pela entidade que disponibiliza a informação clínica. A Aplicação de Visualização deve ser preparada para funcionar num sistema exterior, no qual a entidade que a desenvolveu, não tem qualquer controlo. Isto significa que existe a possibilidade que a aplicação sofra engenharia de reversão, levando a que a implementação fique disponível para terceiros. Estes poderão utilizar o conhecimento adquirido de modo a ludibriar e alterar os mecanismos de segurança implementados. Para tentar dificultar a compreensão e modificação da solução, técnicas, como as de ofuscação[12] e assinatura digital da solução, devem ser utilizadas. É imperativo reforçar que a Aplicação de Visualização apenas cria uma barreira de proteção sobre a informação clínica. Não existe proteção total da informação.

A Aplicação de Visualização poderá receber dois tipos de pedidos oriundos da Aplicação Terceira.

- Pedido para requisitar ao utilizador/detentor um acesso para aceder e/ou manipular a informação clínica em seu nome;
- Pedido para mostra informação clínica ao utilizador/detentor.

[12] Um ato deliberado com o objetivo de dificultar a compreensão do código fonte de uma aplicação

Na requisição de um acesso ao utilizador/detentor, a Aplicação de Visualização irá solicitar ao utilizador detentor que se autentique e que autorize o acesso (Esquema 1 - c)). O processo de autorização de acesso utiliza as credencias da Aplicação Terceira que foram passada no pedido, assim como, as do utilizador/detentor. O processo em si segue o protocolo OAuth1 (2.3.1) para adquirir o acesso (Ilustração 3). O OAuth1 faz uso do protocolo HTTP para enviar mensagens as interações identificadas no Esquema 1 - e) e Esquema 1 - d). Para além disso, o OAuth1 define um conjunto de passos para a aquisição de acesso. Em cada passo mensagens com formatos específicos serão trocadas. Apesar disso todas têm uma característica em comum. Todas as mensagens enviadas são digitalmente assinadas. O protocolo define três modos de assinatura.

- HMAC-SHA1 significa que será utilizada uma função de *hashing*[13] SHA1 e que o resultado será encriptado utilizando uma chave secreta simétrica;
- RSA-SHA1 significa que será utilizada uma função de *hashing* SHA1 mas para a assinar o resultado, será utilizada uma chave assimétrica;
- PLAINTEXT será enviado a chave utilizada, em vez de uma assinatura.

Dependendo da implementação, o modo como é realizada a assinatura pode variar. É importante referir que dependo do passo para adquirir o acesso, a chave utilizada para assinar as mensagens poderá ser diferente. Mais detalhes na especificação do protocolo OAuth1.

No caso das mensagens trocadas, apenas será referido o formato utilizado após a aquisição do acesso. A razão de tal tem a ver com a especificidade de cada mensagem OAuth1, de modo a adquirir o acesso. Dado isso, a seguinte tabela ilustra o formato da mensagem definido na especificação[14] do protocolo OAuth1, para a Aplicação de Visualização poder realizar interações com o exterior.

**Tabela 3 - Formato das mensagens OAuth1 após ter adquirido o acesso**

| | |
|---|---|
| OAuth realm | oauth_consumer_key |
| oauth_token | oauth_signature_method |
| oauth_timestamp | oauth_nonce |
| oauth_signature | |

[13] O *hashing* consiste numa função que gera um conjunto aleatório de caracteres que dependem do conteúdo utilizado. É possível gerar o mesmo conjunto de caracteres se for utilizado o mesmo conteúdo sem alterações. Uma mínima alteração no conteúdo, altera o conjunto aleatório de caracteres.
[14] https://tools.ietf.org/html/rfc5849

O seguinte diagrama ilustra a sequência de interações de modo a adquirir o acesso OAuth1. Neste diagrama foi utilizado a interação com Servidor de Permissões (Esquema 1 - d)) para adquirir o acesso, no entanto, a sequência não mudaria caso o Servidor de Dados (Esquema 1 - e)) tivesse sido escolhido.

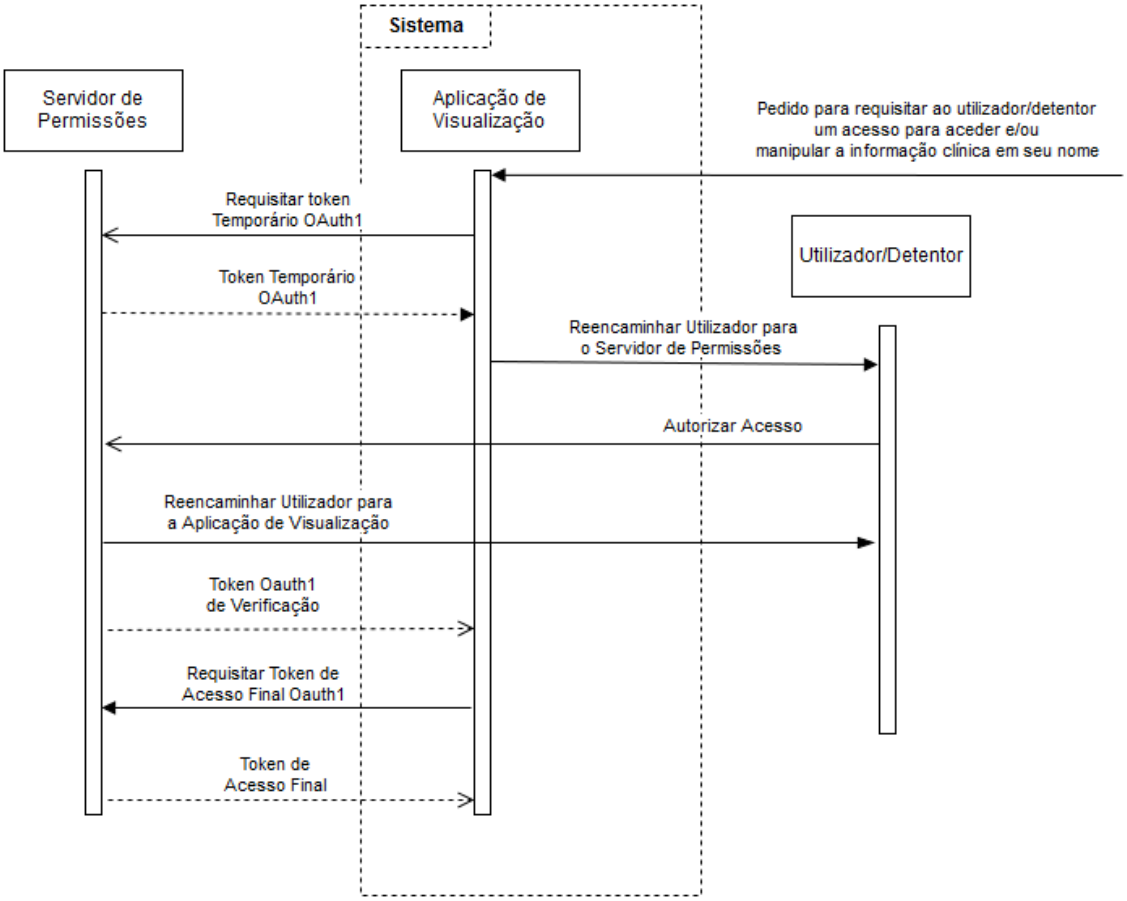



**Ilustração 3 - Interações para processar o pedido para requisitar ao utilizador/detentor um acesso para aceder e/ou manipular a informação clínica em seu nome**

Quando a Aplicação de Visualização recebe o pedido para mostrar informação clinica, assumindo que a Aplicação de Visualização já tenha adquirido o acesso, esta tem que autenticar a proveniência do pedido no sistema. Para tal, as credenciais da Aplicação Terceira devem estar presentes nesse pedido. Deste modo, a Aplicação de Visualização apenas necessita de comparar as credências que recebeu, com as credencias das aplicações terceiras associados os acessos já adquiridos. Caso não se verifique, a aplicação não deve executar o pedido.

Confirmada a autenticação do pedido, a Aplicação de Visualização irá apresentar ao utilizador/detentor a informação clínica. Isto irá permitir que o utilizador possa navegar pela informação clínica, assim como, manipula-la (Esquema 1 - c)).

A informação apresentada, poderá ter origem no sistema onde a Aplicação de Visualização está em funcionamento, isto porque poderá ter sido armazenada no sistema de modo a permitir a consulta *off-line*, para *cache*, entre outras. Se isso acontecer, poderá significar que outras aplicações consigam aceder à informação clínica armazenada sem autorização. Para evitar isso, é imperativo que a informação seja armazenada no sistema, de forma encriptada, como referenciado pelo IRM (2.4). A Aplicação de Visualização utilizará uma chave de desencriptação adquirida na interação Esquema 1 - d) via protocolo HTTP, para desencriptar informação clínica e apresentá-la ao utilizador/detentor. No caso da informação clínica não existir no sistema onde a Aplicação de Visualização está em funcionamento, esta poderá ser adquirida através da interação Esquema 1 - e) via protocolo HTTP, utilizando o acesso anteriormente adquirido. É importante referir que a informação é partilhada de forma encriptada. A aplicação de visualização terá de adquirir a chave de desencriptação igualmente na interação Esquema 1 - d).

É imperativo que chave de desencriptação adquirida, as credenciais recebidas das aplicações terceiras, os acessos à informação clínica e a informação clínica desencriptada, nunca sejam armazenadas no sistema. Estas devem permanecer na memória da Aplicação de Visualização e devem ser corretamente destruídas, de modo a limpar a posição de memória correspondente. O modo como isto é realizado está ligado à linguagem de programação utilizada para desenvolver a Aplicação de Visualização. Esta poderá dispor de mecanismos que permitem a limpeza. Isto assume que o sistema onde Aplicação de Visualização está em funcionamento, implementa mecanismos que evitam acessos indevidos à memória de outras aplicações enquanto estas então em funcionamento. Caso não se verifique, é necessário investigar como implementar outros mecanismos. Apesar de ser necessário este tipo de proteção, é importante referir que este tema sai do âmbito desta dissertação.

Ao permitir que o utilizador/detentor navegue pela informação clínica, torna-se imperativo notificar a Aplicação Terceira sobre que informação clínica está a ser mostrada (Esquema 1 - b)). Isto é importante porque permite que a Aplicação Terceira possa delegar outras possíveis ações sobre a informação que está a ser apresentada. Isto implica que toda a informação clínica tenha um identificador publico, um URI (2.2.2), que possa ser partilhado entre a Aplicação Terceira e de Visualização e comunicada através de IPC's.

Uma das principais funções da Aplicação de Visualização consiste em controlar as operações de manipulação que o utilizador/detentor pode realizar sobre a informação clínica Esquema 1 - c). As operações de manipulação poderão consistir nas operações de cópia, captura de ecrã, impressão, entre outras. Estas operações estão sujeitas a uma confirmação exterior através da interação.

De modo a confirmar as autorizações para as operações de manipulação, assim como, adquirir a chave de desencriptação da informação clínica, a Aplicação de Visualização deverá utilizar a interação Esquema 1 - d) para comunicar com o Servidor de Permissões, de modo a definir nos elementos gráficos a permissão para a operação de manipulação.

### 3.1.4 Servidor de Permissões

O Servidor de Permissões consiste no servidor que disponibiliza os recursos que contém as permissões sobre que operações de manipulação podem ser realizadas sobre a informação clínica, assim como, a chave de desencriptação para o conteúdo disponibilizado na interação Esquema 1 - e). Os recursos são disponibilizados através de um *Web Service* baseado em REST (2.2.2) e podem ser consultadas de duas formas: um URI para cada operação existente, respondendo com a respetiva permissão; ou um URI que responde com todas as operações existentes e respetivas permissões. A Aplicação de Visualização deve ser adaptada consoante a forma escolhida.

Para aceder aos recursos disponibilizados pelo Servidor de Permissões, é utilizado protocolo OAuth1 (2.3.1). É importante referir que o Servidor de Permissões dará apenas permissões de consulta dos recursos. A permissão para a sua edição é atribuída a uma outra entidade responsável por tal. Nesta dissertação, essa entidade não foi definida.

Cada recurso disponibilizado pelo Servidor de Permissões está associado a um utilizador/detentor específico. O conteúdo dos recursos poderá conter um conjunto de permissões de manipulação que podem incluir a captura de ecrã, impressão, copia, ou qualquer outra permissão que as regras de negócio assim o exigem. Esse conteúdo poderá variar, se necessário, consoante a aplicação utilizada, rede, entre outros. Cabe à Aplicação de Visualização interpretar e forçar o cumprimento da permissão. É imperativo referir que qualquer pedido ou requisição de acesso deve ser registado, independente mente do seu sucesso ou não. Esse registo deve conter, se possível, toda a informação necessária para identificar quem realizou o pedido assim como a sua origem.

Para além da disponibilização para o exterior de recursos em que o conteúdo representam permissões, o Servidor de Permissões também disponibiliza acesso à chave para a desencriptação da informação clínica disponibilizada pela interação Esquema 1 - e). Para tal, o servidor de permissões, utiliza a interação Esquema 1 - f), para comunicar com um servidor denominado de Servidor de Dados, de modo adquirir a chave em questão. É importante referir que também pela interação Esquema 1 - f), é partilhado entre o Servidor de Permissões e de dados, os acessos já atribuídos às aplicações terceiras, de modo a evitar a dupla aquisição de acesso.

### 3.1.5 Servidor de Dados

O Servidor de Dados representa o sistema hospitalar que disponibiliza a informação clínica encriptada para o exterior. Este faz uso dos mecanismos e recursos norma definidos na *Framework* de interoperabilidade FHIR (2.1.2). Isto implica que a informação clínica seja disponibilizada através de *Web Services* baseados em REST (2.2.2) e que o protocolo de pedidos e requisições de acesso seja o OAuth1 (2.3.1).

Em relação aos pedidos e requisições de acesso, o servidor de dados deve ter em consideração, não só a permissões associadas ao utilizar/detentor, como as permissões associadas à Aplicação Terceira na qual o pedido está associado. O Servidor de Dados poderá utilizar qualquer mecanismo para representar e aplicar o controlo. É imperativo referir que qualquer pedido e requisições de acesso, dever ser registado, quer tenha sucesso ou não. O registo deve incluir, se possível, toda informação possível de recolher para identificar quem realizou o pedido assim como a sua origem. No caso de um pedido de acesso ser negado, o Servidor de Dados não deverá enviar qualquer informação clínica, mesmo que esta seja encriptada.

Quando é disponibilizada informação clínica para o exterior, o Servidor de Dados encripta essa informação, utilizando uma chave gerada especificamente para o acesso (utilizador/detentor e Aplicação Terceira) correspondente. Esta chave de encriptação poderá ser simétrica ou assimétrica e deve utilizar as especificações de segurança recomendáveis. A chave poderá ser gerada em dois momentos destintos: no momento do pedido de acesso à informação (Esquema 1 - e)); ou no momento em que o Servidor de Permissões requisita a chave (Esquema 1 - f)), no caso da chave ainda não ter sido gerada.

A razão da encriptação da informação clínica, antes da disponibilização, tem como base o funcionamento do IRM (2.4). Ao disponibilizar informação clínica encriptada, é criada uma barreira de segurança, na forma de encriptação pelo canal de comunicação. Deste modo dificulta e protege, até certo ponto, a interceção da informação disponibilizada.

O seguinte diagrama de sequência tem o intuito de ilustrar as interações descritas desde o momento que Aplicação de Visualização recebe um pedido para mostrar informação clínica ao utilizador/detentor, até receber a resposta a esse pedido, passando pelo Servidor de Permissões e de Dados.

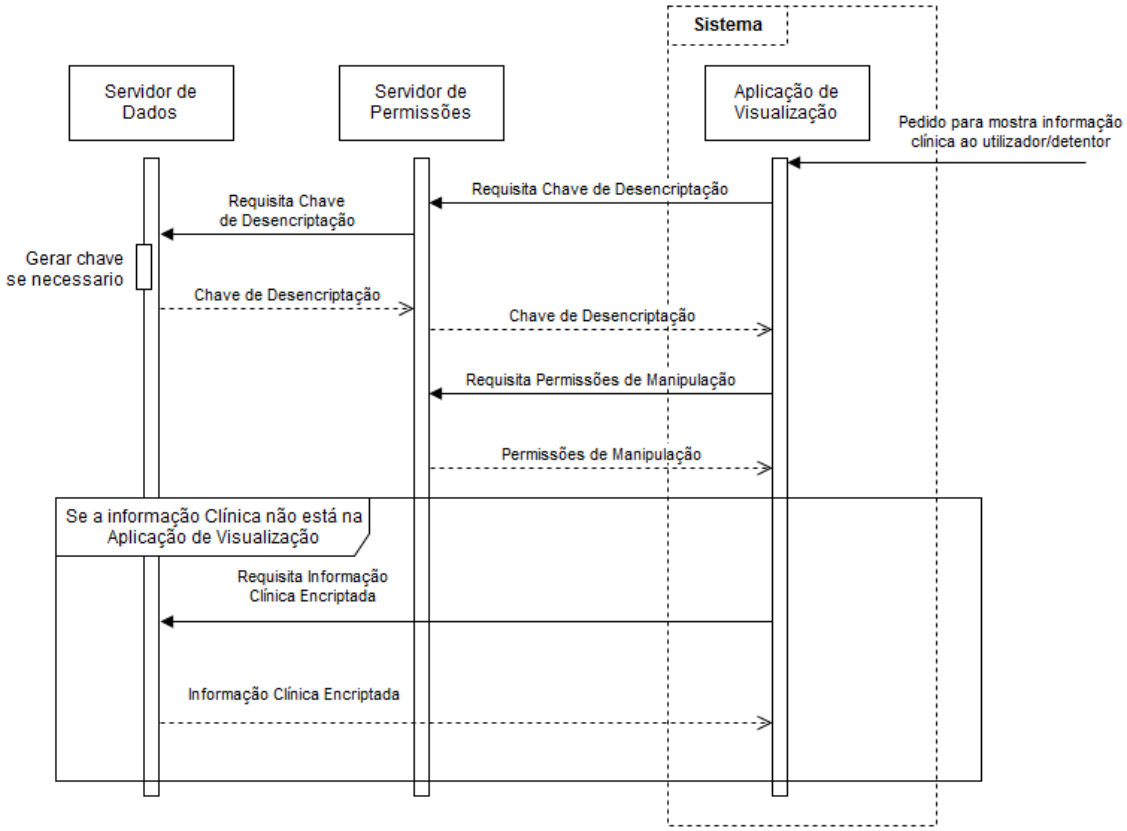



**Ilustração 4 - Interações para processar o pedido para mostra informação clínica ao utilizador/detentor**

# Capítulo 4

# Implementação

Tendo como base o Modelo Conceptual apresentado, o presente capítulo apresenta uma implementação desse modelo. Este tem o intuito de criar uma prova de conceito de modo a poder demostrar e validar a presente dissertação.

No Modelo Conceptual foram definidos quatro componentes: um Servidor de Dados e de Permissões; uma Aplicação Terceira e uma Aplicação de Visualização. É importante referir que nesta implementação os Servidores de Dados e Permissões foram desenvolvidos utilizando a tecnologia C#.Net, enquanto as Aplicações de Visualização e Terceira foram desenvolvidas utilizando o Java versão 8. Estas aplicações inserem-se num contexto de utilização em dispositivos móveis.

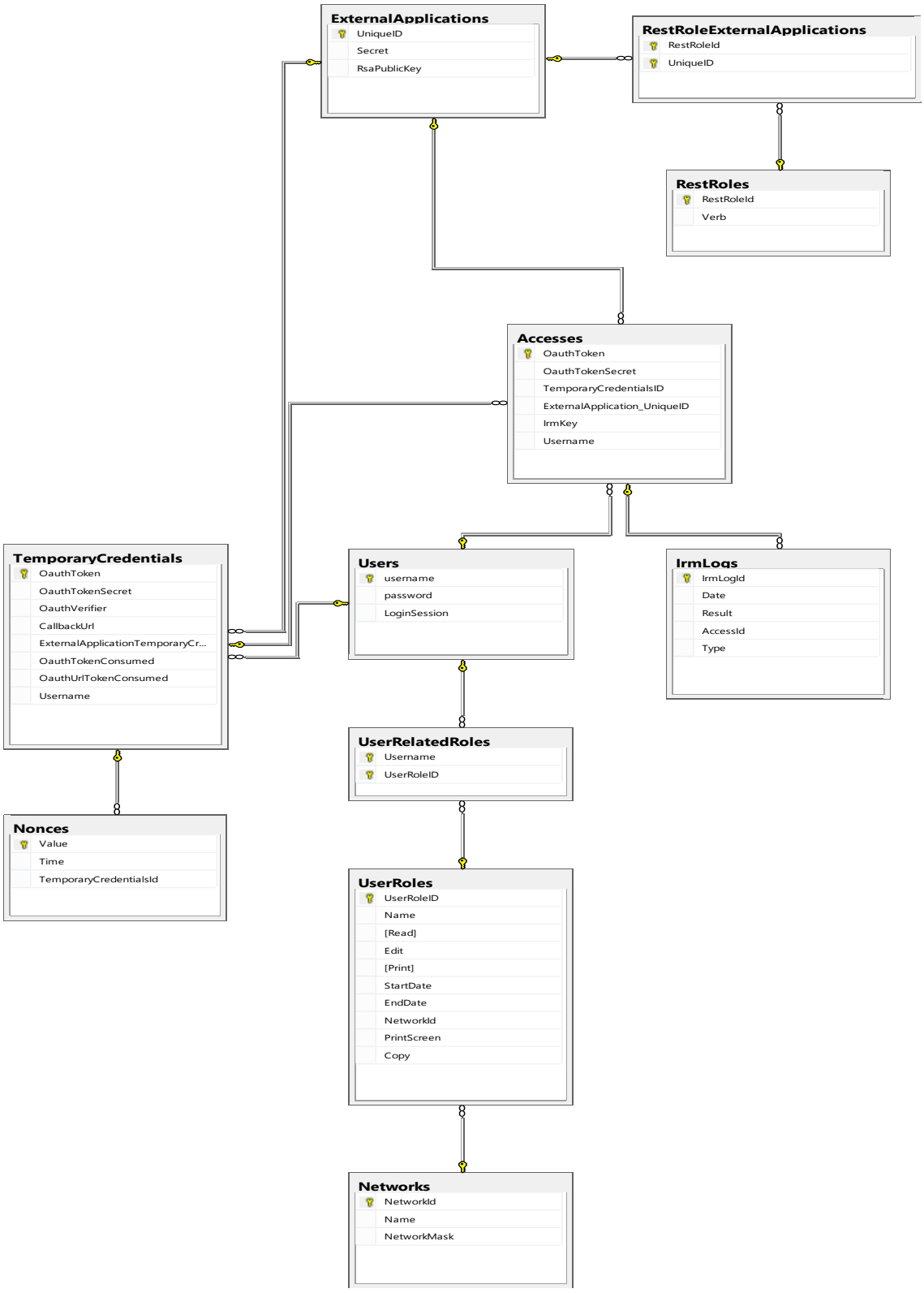
## 4.1 Servidores

Como foi referido no capítulo Análise, Desenho e Arquitetura, o Servidor de Dados e de Permissões têm responsabilidades diferentes, no entanto, ambos partilham informação entre si. Na implementação de cada servidor, o modo como estes comunicam entre si é feito através da base de dados. Por outras palavras, ambos partilham a mesma base de dados.

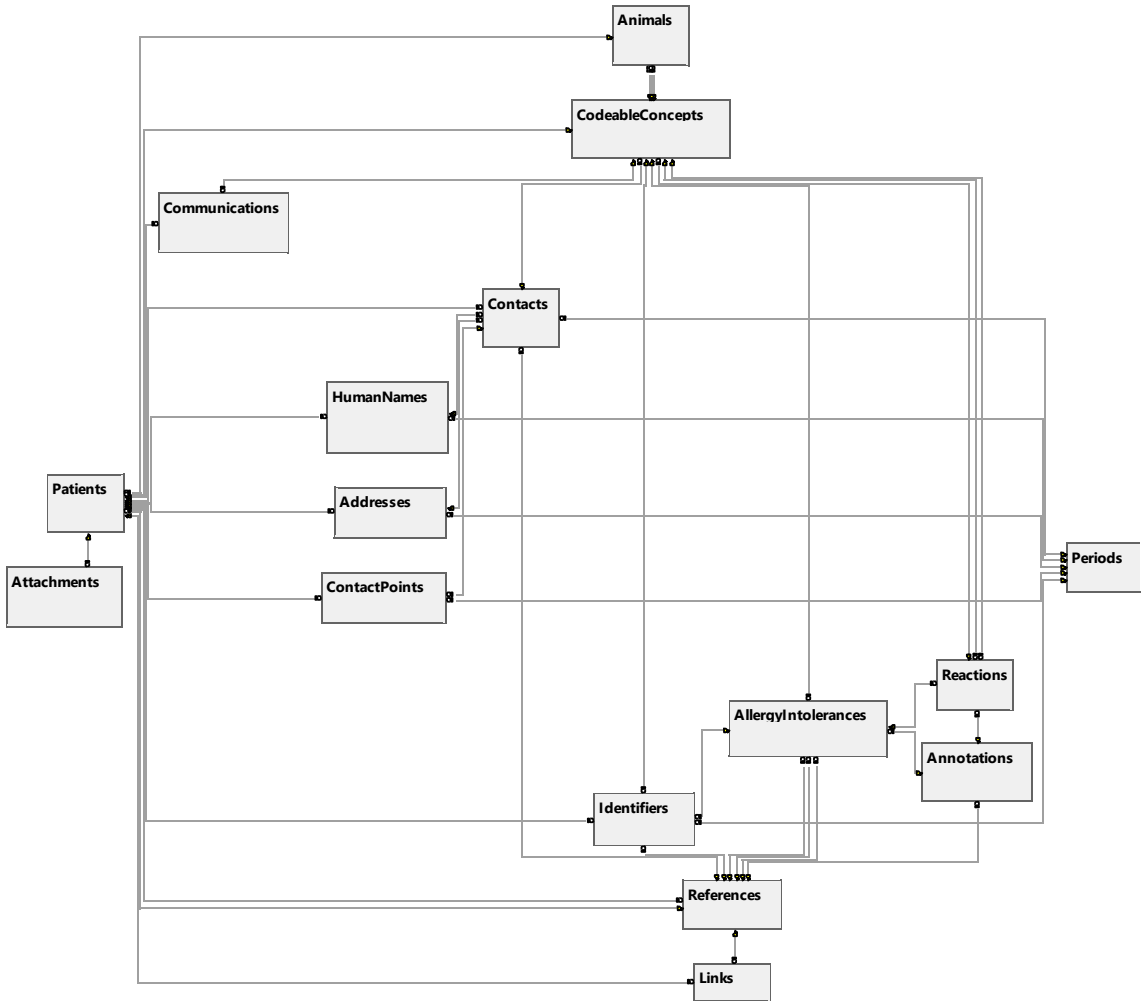
### 4.1.1 Base de Dados

O tipo de base de dados utilizada para representar toda a informação consiste numa base de dados relacional. Esta foi desenvolvida utilizando o Microsoft SQL *Server* como DBMS (*Database Management System*).

A base de dados contém todas as tabelas necessárias para o Servidor de Dados e de Permissões. Isto significa que a base de dados irá conter uma enorme quantidade de tabelas. Apesar disso, é possível agrupar as tabelas em dois grupos destintos: o grupo responsável pelas permissões dos vários intervenientes, gestão dos acessos atribuídos via OAuth1 e pelos registos de atividade (Esquema 2); e o grupo responsável por representar a informação clínica (Esquema 3). Este último grupo tem como base o modo como os recursos FHIR são organizados. Devido à grande quantidade de recursos que o FHIR disponibiliza, foi apenas considerado o recurso do tipo *Patient* e do tipo *AllergyIntolerance*. A razão de ter sido estes dois recursos e não outros, tem a ver com a relação que existe entre eles, no sentido de partilharem tabelas de base de dados. As tabelas referentes aos recursos são apresentadas de modo resumido, isto porque na definição FHIR desses recursos, existe uma enorme quantidade de campos para cada tabela.



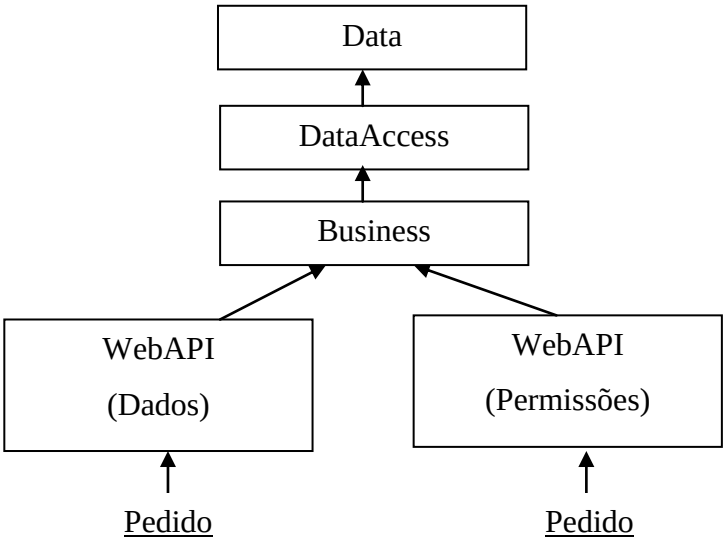

**Esquema 2 - Tabelas da Base de Dados que são responsáveis pelas permissões de acesso, acesso Oauth1 e registos de atividades**




**Esquema 3 - Tabelas da Base de Dados simplificadas com a representação dos recursos FHIR *Patient* e *AllergyIntolerance***

### 4.1.2 Arquitetura Interna

Tanto o Servidor de Dados como o Servidor de Permissões foram desenvolvidos utilizando uma arquitetura interna semelhante. Essa arquitetura é denominada de arquitetura em camadas. Nesta arquitetura, cada camada deve ser autossuficiente, de modo a que seja possível a sua reutilização e/ou distribuição pela rede, se necessário. No caso dos Servidores de Dados e Permissões, foram definidos quatro camadas destintas (Esquema 4).




**Esquema 4 - Esquema da arquitetura interna dos Servidores de Dados e de Permissões**

A camada *Data* consiste na camada que contém a representação em forma de objetos de todas as tabelas presentes na base de dados e respetivas relações. As relações entre os objetos e as tabelas da base de dados foram definidas utilizando algo denominado *de Entity Framework* (EF). A EF consiste num ORM (*Object/Relational Mapping*) desenvolvido pela Microsoft com o intuito de permitir manipulação de informação presente na base de dados através dos objetos e suas instâncias (Entity Framework Tutorial, 2016).

A EF dispõe de uma característica que permitiu a rápida criação das várias tabelas da base de dados. Essa característica tem a denominação de *Code-First*. O *Code-First* permite gerar e manipular as tabelas na base de dados através das alterações realizadas nos objetos.

A camada *DataAccess* contém todas as perguntas (*queries*) SQL disponíveis para manipular a informação guardada na base de dados. O modo como são realizadas as perguntas SQL segue o modo de funcionamento da *Entity Framework*. Isto implica que a camada *DataAccess* conheça e tenha acesso à camada *Data*, de modo a poder aceder aos objetos que representam as tabelas da base de dados.

A camada *Business* contém as regras de negócio que o Servidor de Dados e de Permissões têm que respeitar. Estas foram especificadas no capítulo Análise, Desenho e Arquitetura e fundamentalmente consistem nas seguintes:

- Gerar a chave de encriptação para recursos FHIR disponibilizados e encriptá-los;
- Executar todas as diligências do protocolo OAuth1, assim como, verificar se o utilizador/detentor tem acesso aos recursos requeridos. É a camada *Business* que implementa os mecanismos para controlar o acesso;
- Executar, utilizando os objetos das camadas *DataAccess* e o *Data*, todas as perguntas SQL de modo a extrair ou alterar a informação da base de dados;
- Registar todos os pedidos realizados aos servidores.

É importante referir que a camada *Business* faz uso das bibliotecas criptográficas presentes no C#.Net para gerar a chave de encriptação e para realizar as diligências necessárias do protocolo OAuth1. Para além de utilizar essas bibliotecas, a camada *Business* tem acesso e conhece as camadas *DataAccess* e *Data.*

A camada *Business* contém três características importantes de modo a executar corretamente as regras de negócio especificadas no capítulo Análise, Desenho e Arquitetura:

- A primeira característica consiste em mapear a informação recebida da camada *WebAPI* para um formato definido na camada *Data*, de modo a que haja compatibilidade com as funcionalidades existentes;
- A segunda característica consiste na criação de uma transação, com o devido nível de isolamento, cada vez que executa uma regra de negócio. Isto é importante para manter a integridade da informação na base de dados;
- A terceira característica tem a ver com o tratamento de erros. Durante a execução de uma regra de negócio, exceções podem acontecer e é necessário tratá-las devidamente, antes de passar a resposta para a camada *WebAPI*.

A camada *WebAPI* consiste na camada de entrada para o Servidor de Dados e para o Servidor de Permissões. Ao contrário das camadas referidas anteriormente, a camada *WebAPI* conhece apenas a camada *Business*. A camada *WebAPI* foi desenvolvida utilizando o C#.Net Web API, que consiste numa *Framework* com o intuito de facilitar o desenvolvimento de *web services*, tais como os baseados em RESTful (Microsoft, 2016).

O C#.Net Web API segue uma arquitetura baseada em MVC (*Model-View-Controller)*. O MVC consiste numa arquitetura em que divide uma aplicação três componentes: *Model*, *View* e *Controller* (Tutorials Point, 2016):

- O *Model* representa toda a lógica por de traz dos dados que a aplicação vai utilizar;
- A *View*, no qual contem os componentes gráficos que serão apresentados aos utilizadores;
- O *Controller, que* consiste na interface entre o *Model* e a *View*. É o *Controller* que recebe todos os pedidos oriundos do exterior.

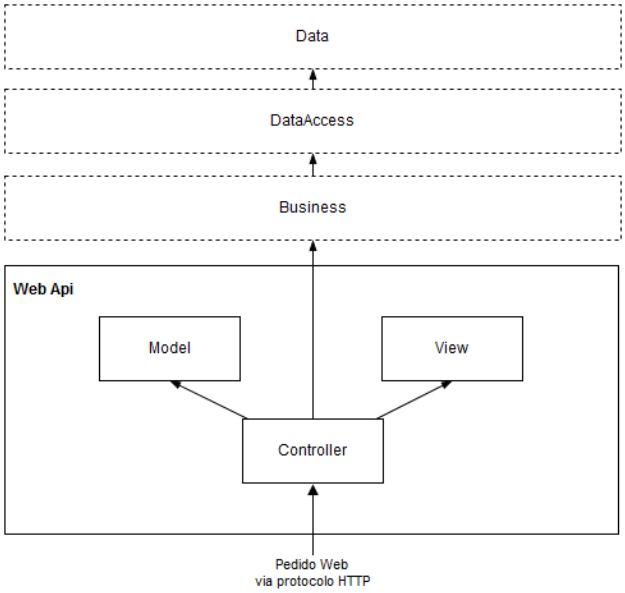



**Ilustração 5 - Arquitetura MVC (Model, View e Controller) de um WebApi**

No caso do Servidor de Dados e de Permissões, só o *Controller* será utilizado. Este terá a intuito de invocar as funcionalidades presentes na camada *Business* mais adequadas ao pedido, assim como, proporcionar a resposta mais adequada no formato mais adequado consoante o resultado da operação executada na camada *Business.*

O Servidor de Dados e de Permissões, para além de partilharem a base de dados, partilham a implementação presente nas camadas *Data*, *DataAccess* e *Business*. Só na camada *WebAPI* é que não é partilhada. Cada um tem a sua própria camada de *WebAPI.* Isto significa que cada servidor contém um domínio local próprio no qual disponibiliza as suas funções. Estas podem ser acedidas via RESTful. Cada função disponibilizada e denominada de *endpoint*. A razão de tal tem a ver com o facto de cada servidor oferecer funções diferentes.

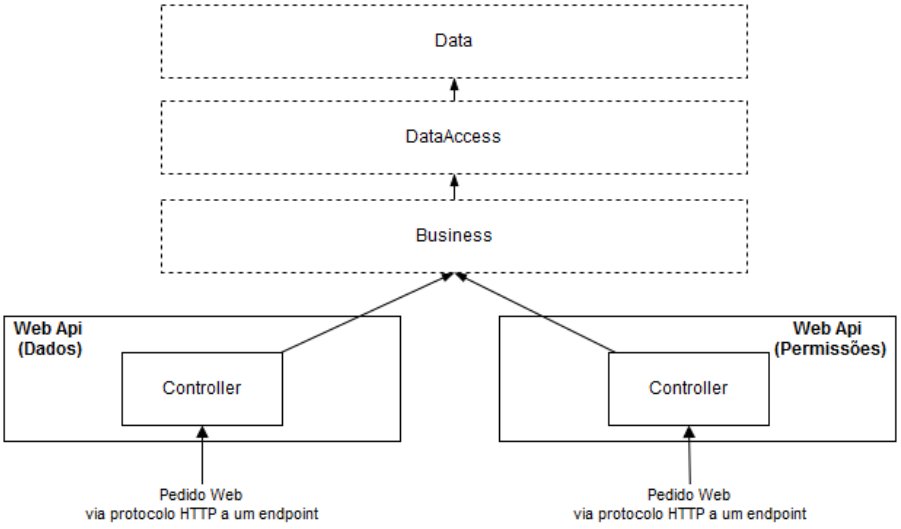



**Ilustração 6 - Arquitetura interna dos servidores em detalhe**

As seguintes tabelas de *endpoints* indicam as funções específicas de cada servidor.

As seguintes tabelas indicam os *endpoints* existentes na camada *WebAPI* do Servidor de Permissões.

**Tabela 4 - *EndPoints* para a iniciação da aquisição de acesso via OAuth1**

| API | Description |
|---|---|
| POST api/Initiate | Inicia a requisição de acesso OAuth1.<br>Deve incluir as credenciais da aplicação que quer aceder à informação.<br>Retorna um *token* temporário que serve para redirecionar o utilizador/detentor e para identificar o pedido de acesso |

**Tabela 5 - *EndPoints* para a realização do login do utilizador/detentor**

| API | Description |
|---|---|
| GET api/Login | Devolve os CSRF[15] *tokens* associados para a submissão do pedido do utilizador/detentor<br>Deve incluir um *token* temporário. |
| POST api/Login | Submissão da informação para o login do utilizador/detentor.<br>Deve incluir todos os CSRF *tokens* já adquiridos, assim como, o *token* temporário.<br>Se a informação de login estiver correta, devolve um *cookie* de sessão. |

[15] O CSRF (*Cross-Site Request Forgery*) consiste num tipo de ataque informático com o intuito de submeter um formulário para executar uma ação, sem que o utilizador tenha requerido tal formulário para executar a ação.

**Tabela 6 - *EndPoints* para indicar a autorização de acesso**

| API | Description |
|---|---|
| GET api/Authorize | Devolve os CSRF *tokens* associados para a submissão do pedido do utilizador/detentor.<br>O pedido deve inclui o *cookie* de sessão adquirido no login, assim como, o *token* temporário |
| POST api/Authorize | O utilizador/detentor confirma a autorização de acesso.<br>Deve ser enviado os CRSF *tokens*, assim como, o *cookie* de sessão proveniente do login e o *token* temporário<br>É retornado um *token* confirmação de autorização OAuth1. |

**Tabela 7 - *EndPoint* para aquisição do *token* de acesso final OAuth1**

| API | Description |
|---|---|
| POST api/Token | Requisita o *token* final de acesso OAuth1.<br>Este deve incluir o *token* de autorização de confirmação<br>É retornado o *token* de acesso final OAuth1 |

**Tabela 8 - *Endpoints* para consulta das permissões IRM do utilizador**

| API | Description |
|---|---|
| GET api/Irm/Key | Obtenção da chave de desencriptação do conteúdo.<br>Necessário usar o *token* de acesso final OAuth1. |
| GET api/Irm/Print | Consulta da permissão para impressão.<br>Necessário usar o *token* de acesso final OAuth1. |
| GET api/Irm/PrintScreen | Consulta da permissão para captura de ecrã.<br>Necessário usar o *token* de acesso final OAuth1. |
| GET api/Irm/Copy | Consulta da permissão para *copy paste* do conteúdo.<br>Necessário usar o *token* de acesso final OAuth1. |

As seguintes tabelas indicam os *endpoints* existentes na camada *WebAPI* do Servidor de Dados. Como foi referido, a informação clínica é disponibilizada de forma encriptada. Para tal foi utilizado na camada *Business* bibliotecas criptográficas C#.Net. Dos algoritmos disponíveis nessas bibliotecas, foi utilizado o AES (*Advanced Encryption Standard*) em modo CBC (*Cipher Block Chaining*). Isto significa que a informação é encriptada em blocos e que cada bloco é encriptado tendo em conta o bloco encriptado anterior. No caso do primeiro bloco, como não existe bloco anterior, foi definido um vetor de inicialização.

**Tabela 9 - *Endpoints* do *AllergyIntolerance***

| API | Description |
|---|---|
| GET api/AllergyIntolerance | Devolve a lista de recursos FHIR denominados de *AllergyIntolerance*.<br>Necessário usar o *token* de acesso final OAuth1. |
| GET api/AllergyIntolerance/{id} | Devolve uma *AllergyIntolerance* especificada pelo parâmetro {id}<br>Necessário usar o *token* de acesso final OAuth1. |

**Tabela 10 - *EndPoints* do *Patient***

| API | Description |
|---|---|
| GET api/Patient | Devolve a lista de recursos FHIR denominados de *Patient.*<br>Necessário usar o *token* de acesso final OAuth1. |
| GET api/Patient/{id} | Devolve uma *Patient* especificada pelo parâmetro {id}.<br>Necessário usar o *token* de acesso final OAuth1. |

### 4.1.3 Permissões de Manipulação e de Acesso à Informação Clínica

O controlo de acesso aos *endpoints* funciona em duas vertentes. Na vertente da aplicação e na vertente do utilizador/detentor. A vertente da aplicação define apenas o tipo de pedido que uma aplicação pode fazer. Por outras palavras, controla se a aplicação pode ou não realizar pedidos do tipo GET, POST, PUT, POST ou OPTIONS aos diversos *endpoints.* Isto aplica-se tanto no Servidor de Dados como no de Permissões. Já na vertente do utilizador/detentor, este possui acesso ilimitado sobre todos *endpoints*. Quando uma aplicação adquire um acesso em nome do utilizador/detentor via OAuth1, a aplicação está limitada apenas pelo tipo de pedido que pode realizar.

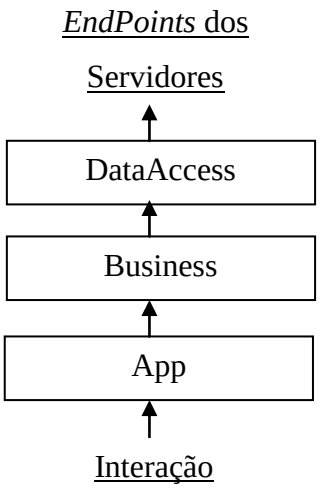
Apesar do utilizador/detentor ter um acesso ilimitado aos *endpoints*, este possui restrições sobre o que pode fazer com a informação que recebeu do Servidor de Dados. As restrições, denominadas de permissões de manipulação, são acedidas no Servidor de Permissões. A atribuição e representação das permissões têm como base o papel do utilizador/detentor na organização, isto é, as permissões de manipulação são atribuídas com base do grupo a que o utilizador/detentor pertence. O utilizador/detentor poderá pertencer a mais que um grupo. É importante referir que cada grupo dispõe do mesmo número de permissões de manipulação. Se todos os grupos a que o utilizador/detentor faz parte permitirem a manipulação requerida, esta é permitida, caso contrário, se algum grupo não a permitir, a manipulação é negada.

## 4.2 Aplicações

De modo a realizar a demostração e posterior validação, foram desenvolvidas duas aplicações: uma aplicação denominada de Visualização e outra designada por Terceira. Estas aplicações foram desenvolvidas tendo um contexto de aplicações móveis, mais especificamente para Android API 23 ou superior.

### 4.2.1 Aplicação de Visualização

A Aplicação de Visualização foi desenvolvida utilizando uma arquitetura semelhante à que foi utilizada nos servidores. Utiliza igualmente uma arquitetura em camadas (Esquema 5).




**Esquema 5 - Arquitetura base das aplicações de Visualização**

Na presente arquitetura, a interação inicia-se com a camada *App*. Esta camada representa a aplicação móvel que vai interagir com o utilizador/detentor. Para tal, a Aplicação de Visualização utiliza algo denominado de atividade (*Activity*). Uma atividade consiste num ecrã apresentado no dispositivo móvel que poderá interagir com o utilizador/detentor (Android Developers, 2016). Este ecrã poderá conter componentes gráficos para apresentar informação ao utilizador/detentor, assim como, permitir a sua manipulação (Android Developers, 2016). As atividades têm a capacidade de comunicarem entre atividade da mesma aplicação e com atividades de outras aplicações (Android Developers, 2016).

A camada *App* é responsável, não só por interagir com o utilizador detentor, como por proteger a informação clínica partilhada (4.2.1.1). Para tal, a camada *App* contem três atividades: a atividade de listagem, de detalhes e de *login*.

- A atividade de listagem irá listar os recursos FHIR, consoante a indicação recebida;
- A atividade de detalhes é invocada quando um item na listagem é selecionado. Nesta irá aparecer toda a informação detalhada do item;
- A atividade de *login* tem o objetivo executar automaticamente a aquisição de acesso OAuth1 quando invocada.

As atividades na camada *App* necessitam de saber quais as regras de negócio que devem seguir para proteger a informação apresentada. Para tal, a camada *App* comunica com a camada *Business,* que corresponde módulo Android. Esta camada é responsável por executar pedidos vindos da camada *App*. Esses pedidos podem ter vários intuitos, tais como:

- Realizar a troca de informação entre os servidores de modo a adquirir um acesso OAuth1;
- Requisitar as permissões de manipulação no Servidor de Permissões sobre o utilizador/detentor atual;
- Requisitar acesso aos recursos FHIR disponibilizados pelo Servidor de Dados, assim como, desencriptá-los. Para tal, irá utilizar o mesmo algoritmo usado pelo servidor para desencriptar, isto é, AES em modo CBC.

É importante referir que para a troca de mensagens OAuth1 e para desencriptar aos recursos FHIR, foi utilizado as bibliotecas criptográficas do Android.

Tal como a camada *Business* nos servidores, a camada *Business* das aplicações também têm algumas características específicas. A seguinte lista enumera quais são essas características:

- Executar os pedidos vindos da camada *App* de modo síncrono ou assíncrono, para evitar o bloqueio da aplicação;
- Tratar devidamente os erros antes de os passar para a camada *App*;
- Mapear a informação oriunda da camada *App* para informação no modelo utilizado na camada *DataAccess* ou *Business*, assim como, mapear a resposta oriunda da camada *DataAccess* para o modelo de informação utilizado pela camada *Business* ou *App*.

A camada *Business* executa um conjunto de ações que têm como objetivo contactar os servidores de Dados e de Permissões. Para tal, a camada *Business* faz uso da camada *DataAccess.* Esta camada consiste num módulo Android que contém os mecanismos necessários para comunicar com os servidores. O contacto com os servidores é realizado através de uma biblioteca presente na API do Android que realiza pedidos HTTP.

Dependo do tipo de atividade invocada, diferentes *endpoints* são invocados pelo *Business* através do *DataAccess.* O seguinte diagrama de sequência ilustra quais serão os *endpoints* invocados pela atividade de login até que a aplicação de visualização adquira o acesso.

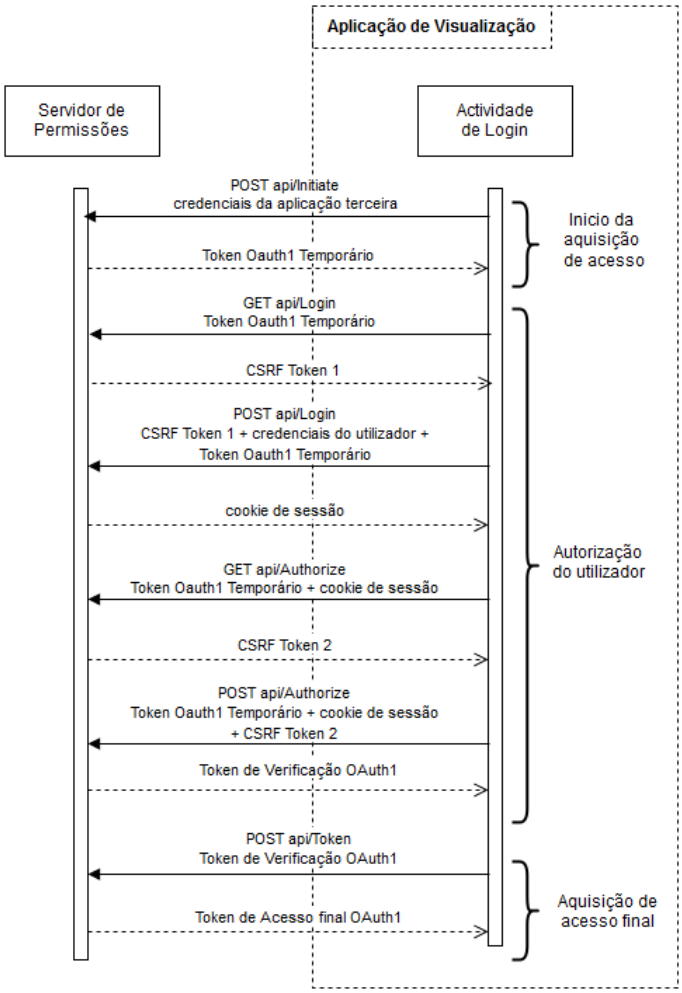



**Ilustração 7 - Sequencia de ações executadas no Business na consequência da atividade de login**

De modo a simplificar a aquisição de acesso para a demostração, a atividade de login executa automaticamente a confirmação de autorização de acesso em nome do utilizador.

O seguinte diagrama de sequência exemplifica uma sequência de ações nas atividades de listagem e de detalhes quando o utilizador/detentor seleciona a lista de *Patient* seguido de detalhe do *Patient*. No caso de ser o recurso FHIR *AllergyIntolerance*, a mesma sequência se aplica, trocando apenas os URI's *Patient* por *AllergyIntolerance.*

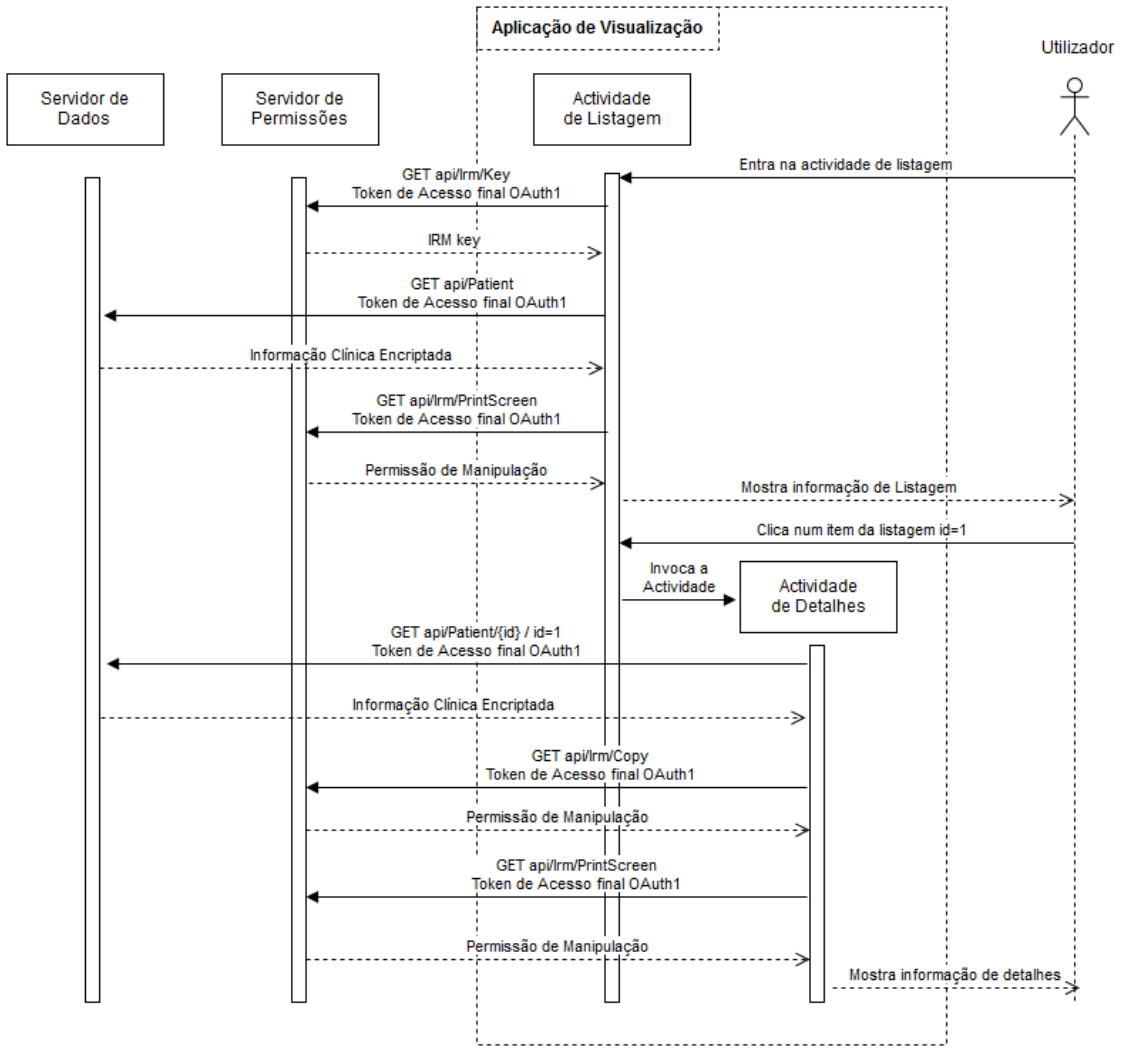



**Ilustração 8 - Sequência de ações executadas no *Business* na consequência da atividade de listagem do *Patient* e da atividade de detalhes do *Patient***

É importante referir que a chave de desencriptação (IRM key), já se encontra na aplicação de visualização quando a atividade de detalhes recebe a informação clínica encriptada. Esta foi adquirida pela atividade de listagem.

#### 4.2.1.1 Mecanismo de Proteção e Controlo da Informação Clínica

Uma das principais responsabilidades das atividades definidas na Aplicação de Visualização consiste em proteger e controlar a manipulação da informação clínica apresentada.

Para poder controlar a manipulação de informação clínica, é necessário definir quais os tipos de manipulação suportados pela Aplicação de Visualização. Na implementação, foram escolhidos dois tipos: a captura de ecrã e a cópia (*copy paste*).

Para controlar as capturas de ecrã, cabe à Aplicação de Visualização decidir, em cada atividade, se bloqueia ou não a sua execução. Por defeito, a atividade permite sempre a execução da captura de ecrã. Para bloquear, as atividades disponibilizam uma etiqueta denominada de "*flag_secure"*. Esta etiqueta está presente em todas as atividades. A Aplicação de Visualização faz uso dessa etiqueta para bloquear a captura de ecrã no momento da criação da atividade. Esta será desbloqueada quando for confirmada a permissão para a sua execução.

O controlo da cópia ("*copy-paste"*) tem objetivo de controlar a seleção e cópia de informação clínica apresentada nos componentes gráficos das atividades. Para tal, foi necessário utilizar um método que todos os componentes gráficos que permitem a seleção têm ao seu dispor. O método consiste na possibilidade de controlar se é ou não permitido selecionar a informação. Dado isso, o controlo é feito através da negação da seleção. A Aplicação de Visualização bloqueia todas a seleções no momento da criação da atividade. Esta será desbloqueada quando a Aplicação de Visualização confirmar que existem permissões para a sua execução.

Para implementar um mecanismo que consiga controla o acesso direto de outras aplicações aos registos clínicos, é necessário ter em conta duas vertentes: O armazenamento no dispositivo e a memória da aplicação. O controlo da informação armazenado no dispositivo é realizado utilizando as técnicas de IRM. Isto significa que esta é encriptada, utilizando a chave adquirida no servidor de permissões, antes do seu armazenamento. Deste modo apenas a Aplicação de Visualização pode consultar, porque apenas ela tem a chave utilizada.

A chave utilizada está na memória da aplicação, assim como, os acessos adquiridos e a informação clinica desencriptada de modo a exibir ao utilizador. A proteção dessa memória cabe ao sistema operativo. Este deve garantir que as aplicações apenas acedem as suas posições de memória. Quando a aplicação é terminada, a sua memória é libertada para poder ser utilizada por outra aplicação. Quando isto acontece, informação sensível utilizada pela aplicação que terminou poderá ficar disponível, como por exemplo, a chave de encriptação da informação clínica. Para evitar tal situação, é necessário ter em conta o modo com a linguagem de programação destrói a informação utilizada na da aplicação.

O Java consiste na linguagem de programação utilizada para a implementação. Esta linguagem é de alto nível e dispõe de algumas características. Uma dessas características consiste no modo como a linguagem permite executar a limpeza da memória. Para tal, é necessário que sejam eliminadas todas as referências para o objeto a ser eliminado e de seguida chamar o GC (*Garbage Collector*). O GC tem o objetivo de eliminar todos os objetos que estejam em memória e que não tenham nenhuma referência. A sua chamada apenas sinaliza que é necessário limpar a memória. É o sistema que escolhe quando este deve ser executado. Isto leva à incerteza sobre se o objeto ainda está ou não em memória.

Na implementação, a sinalização para a limpeza da memória é executada sempre que a aplicação e destruída pelo sistema operativo ou é realizado o *logout*. Quando isso acontece, é eliminado a chave utilizada está na memória da aplicação, os acessos adquiridos e a informação clinica desencriptada.

### 4.2.2 Aplicação Terceira

A Aplicação Terceira representa uma aplicação simples que tem principal objetivo exemplificar como esta poderia interagir com a Aplicação de Visualização. No seu desenvolvimento, não foi utilizado qualquer módulo, isto é, olhando para arquitetura da Aplicação de Visualização, apenas existe a camada *App*.

A camada *App* apresenta apenas duas atividades. Uma para simular um *login* na Aplicação Terceira, e outra que possibilita invocar a atividade na Aplicação de Visualização responsável por mostrar informação clínica ao utilizador/detetor denominada de atividade *Main*. Essa atividade invocada consiste na atividade de listagem.

A atividade que simula o *login* tem a capacidade de invoca a atividade na Aplicação de Visualização para adquirir o acesso OAuth1. Nesta invocação é passado todas as informações necessárias. Após a execução do *login*, o utilizador/detetor é encaminhado para a segunda atividade da Aplicação Terceira.

#### 4.2.2.1 Comunicação entre Aplicações

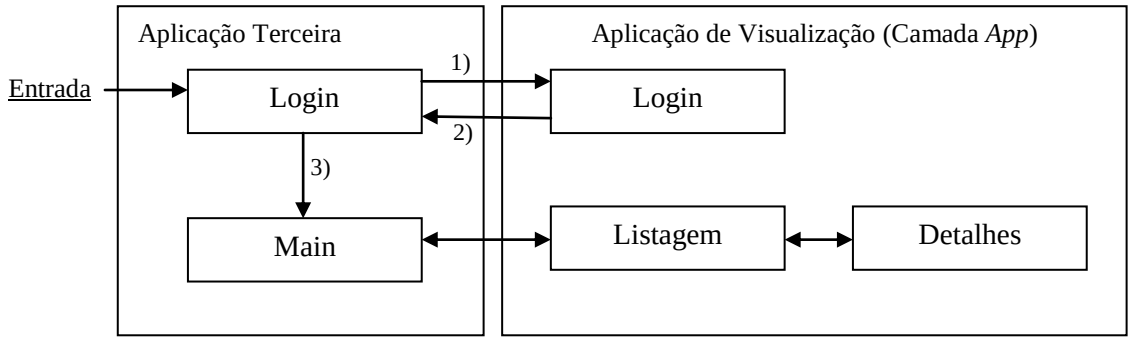
Uma das características das atividades consiste em possibilitar a comunicação entre atividades. Para tal, tanto a Aplicação de Visualização com a Terceira fazem uso de um mecanismo denominado de intenção (*Intent*). A intenção permite invocar atividades, quer na aplicação onde a atividade está definida, quer em outras aplicações. Para tal basta utilizar o identificador único da atividade. No caso de ser uma atividade de outra aplicação, também é necessário indicar o identificador único da aplicação.

A intenção dispõe de algumas características. Uma dela consiste em possibilitar a passagem de informação entre atividades. Esta característica é especialmente útil para a Aplicação Terceira, a quando da delegação de ações para Aplicação de Visualização. A outra característica consiste na capacidade síncrona da intensão, isto é, invocar uma atividade, passando-lhe informação, e esperar pela sua resposta antes de voltar à atividade de origem. Esta capacidade é útil para a Aplicação Terceira, no sentido de ser utilizada para requisitar à Aplicação de Visualização a requisição de acesso OAuth1, voltando depois à Aplicação de Terceira.

Outra característica da proteção da informação clínica consiste o acesso direto à memória. O acesso direto à memória de uma aplicação por outras aplicações está relacionado com o sistema operativo. Cabe a este implementar mecanismos que isolam a memória das aplicações, assim como, disponibilizar mecanismos seguros de comunicação entre aplicações.

### 4.2.3 Esquema de Navegação entre Atividades

De modo a demostrar como as atividades da Aplicação de Visualização e da Aplicação Terceira interagem entre si, é apresentado o seguinte esquema de relações entre atividades. É de notar que algumas relações estão numeradas. Isto significa que essas relações devem ser executadas pela ordem definida automaticamente. As restantes relações não numeradas podem ser exploradas com o utilizado entender.




**Esquema 6 - Navegação entre as atividades da Aplicação Terceira e de Visualização**

## 4.3 Demostração

Esta secção tem o objetivo de apresentar uma utilização da implementação descrita no capítulo anterior. Essa utilização está inserida no seguinte contexto:

Uma clínica pediátrica privada presta cuidados de saúde a crianças. Para registar e consultar os registos realizados, a clínica faz uso de um sistema interno que funciona através de uma aplicação presente nos dispositivos móveis dos profissionais de saúde. Esta aplicação é denominada de Aplicação Terceira. Infelizmente, os profissionais de saúde apenas têm acesso aos cuidados realizados na própria clínica. De modo a auxiliar a prestação de cuidados, é imperativo que os profissionais de saúde obtenham acesso aos registos clínicos externos dos seus pacientes. Para tal, será utilizado uma Aplicação de Visualização desenvolvida pelo sistema que contém esses registos. Esta aplicação tem o objetivo de garantir que os registos clínicos, são utilizados apenas para auxiliar a prestação de cuidados de saúde e não para outras atividades.

Para efeitos da demostração, foram utilizados os recursos FHIR *Patient* e *AllergyIntorange.* Estes não incluem dados reais. Dos dados apresentados, apenas os campos essenciais para a demostração foram considerados. Os restantes encontram-se a nulo. A razão tem a ver com o facto de que a sua representação completa iria ser custoso que não iria trazer diferenças para a demostração.

### 4.3.1 Aquisição de Acesso via OAuth1

Tudo se inicia quando o profissional de saúde abre a Aplicação Terceira para iniciar o seu trabalho. Este terá de inserir as suas credenciais (“*username”* e “*password”*) de acesso à Aplicação Terceira (Ilustração 9). É importante notar que este *login* é apenas demostrativo, isto porque não existe nenhum servidor de *login* para a Aplicação Terceira.

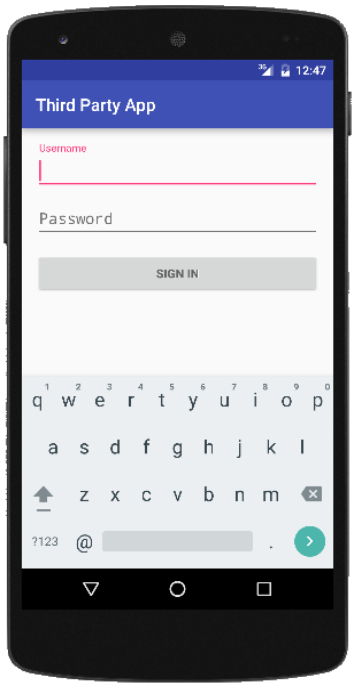



**Ilustração 9 - Login na Aplicação Terceira**

De modo a simplificar a demonstração, as credenciais de acesso à Aplicação Terceira de Aplicação são as mesmas da Aplicação de Visualização. Dado isso, após o *login* demostrativo na Aplicação Terceira (Ilustração 11), é realizado o *login* real na Aplicação de Visualização automaticamente utilizando as mesmas credenciais (Ilustração 10). Estas foram passadas utilizando a Intent .

O *login* real na Aplicação de Visualização irá executar não só o *login*, mas também uma sequência de ações que culminarão com a atribuição de um acesso OAuth1 à Aplicação de Visualização. É importante referir as ações para a aquisição acesso são registadas na tabela de Base de Dados *TemporaryCredentials*. Já o acesso final é registado na tabela *Accesses*.

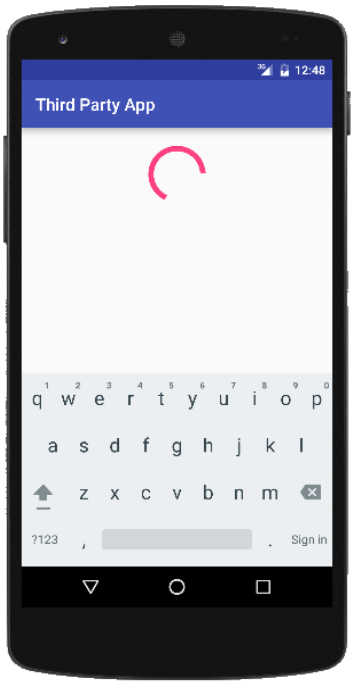



**Ilustração 11 - Execução do *login* demostrativo na Aplicação Terceira**

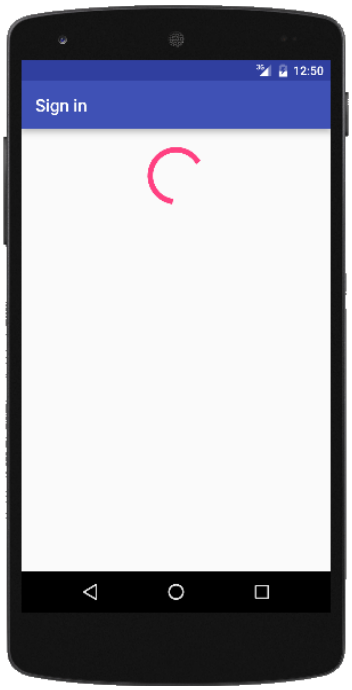



**Ilustração 10 - Aquisição de acesso Oauth1 na Aplicação de Visualização**

Após a execução do *login* real, o profissional de saúde é reencaminhado para o inferior da Aplicação Terceira. Esta é apresentada na Ilustração 12. É importante reforçar que a atividade é apenas demostrativa. Esta tem o objetivo de exemplificar como uma aplicação terceira poderia ser desenvolvida.

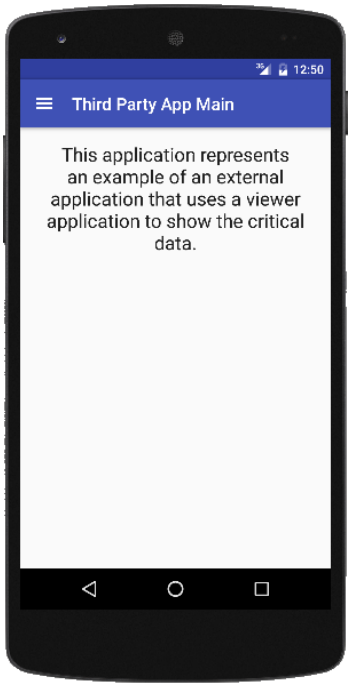



**Ilustração 12 - Aplicação Terceira após *login* real na Aplicação de Visualização**

### 4.3.2 Interação entre Aplicações e Atividades de mesma Aplicação

A interação da Aplicação Terceira com a Aplicação de Visualização é realizada através de menus que permitem invocar as atividades presentes na Aplicação de Visualização. Para tal, a Aplicação Terceira faz uso do mecanismo denominado de intenção (4.2.2.1). É importante referir que, neste caso, apenas foram considerados as atividades responsáveis pela listagem do tipo de recurso FHIR *Patient* e *AllergyIntorange* (Ilustração 13). A razão tem a ver com o facto de que a atividade de detalhes é invocada dentro da Aplicação de Visualização.

Para além das listagens, a Aplicação Terceira contem uma opção de *logout* (Ilustração 13). Esta opção permite limpar e eliminar toda a informação adquirida pelo pela Aplicação de Visualização. Para tal, faz uso dos mecanismos presentes no Java que permitem comunicar com o sistema operativo para limpar a memória.

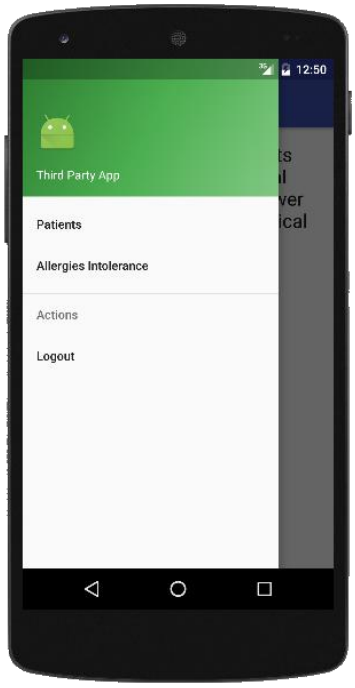



**Ilustração 13 - Menus de navegação na Aplicação Terceira para a Aplicação de Visualização**

Ao selecionar a opção *Patients*, o profissional de saúde irá ser redirecionado para a atividade na Aplicação de Visualização responsável pela listagem (Ilustração 14). O mesmo acontece ao selecionar a opção *AllergyIntorange*. Nessa atividade, um conjunto sequencial de ações exteriores são executadas antes de apresentar a listagem ao profissional de saúde. Essas ações consistem na obtenção da lista de conteúdo de um servidor RESTful, a chave desencriptação do conteúdo e as permissões de captura de ecrã do profissional de saúde.

É de notar que existe uma exceção nas ações. A obtenção da chave de desencriptação do conteúdo e do conteúdo em si é executado de modo síncrono. Por outras palavras, só após receber a informação com sucesso é que é apresentada a listagem. Já a aquisição da permissão de captura de ecrã, é realizada de modo assíncrono. Isto significa que a atividade é criada e apresentada ao profissional de saúde com a permissão de captura de ecrã bloqueada. Esta será desbloqueada quando houver confirmação que tal manipulação é possível. Todas as requisições de consulta ao Servidor de Dados serão guardadas na tabela de Base de Dados *IrmLogs*. Inclui igualmente se o resultado do pedido foi executado, ou não, com sucesso.

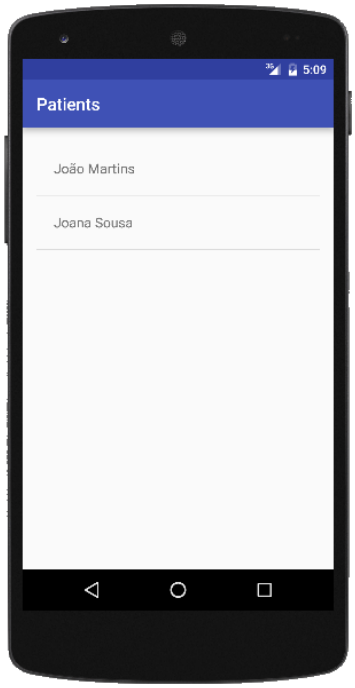



**Ilustração 14 - Listagem do recurso FHIR *Patient* na Aplicação de Visualização**

A atividade de listagem na Aplicação de Visualização permite, através do clique num item, consultar detalhes sobre o *Patient* (Ilustração 15). Quando tal acontecer, uma atividade na Aplicação de Visualização irá ser invocada através de uma intenção (4.2.2.1). Essa atividade é responsável por mostrar os detalhes do item selecionado. É importante notar que na implementação, não foi considerado a notificação à Aplicação Terceira que um item foi clicado, devido ao facto de a Aplicação Terceira ser bastante simples.

Ao obter a informação encriptada do *Patient*, é importante referir que a chave de desencriptação já se encontra na Aplicação de Visualização. Esta foi adquirida na atividade de listagem.

À semelhança da atividade de listagem, a obtenção da informação sobre o *Patient* é realizada de modo síncrono a um servidor RESTful, isto é, só após a obtenção da informação é que a mesma é apresentada. Já as consultas das permissões de manipulação de captura de ecrã e cópia ("*copy-paste"*), são realizadas de modo assíncrono, isto é, são bloqueadas até que sejam confirmadas as autorizações de manipulação.

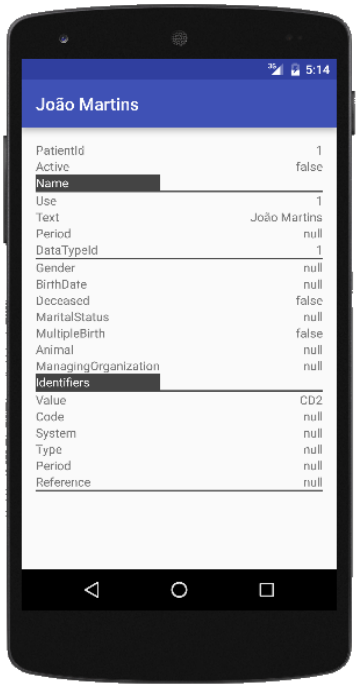



**Ilustração 15- Detalhe do Paciente na Aplicação de Visualização**

### 4.3.3 Ações de Manipulação

Um profissional de saúde poderá executar dois tipos de operações de manipulação. Captura de ecrã e cópia ("*copy-paste"*). É importante referir que todas estas consultas de manipulação realizadas ao Servidor de Permissões, assim como se o seu resultado teve ou não sucesso, são registadas na tabela de Base de Dados *IrmLogs*.

Antes da execução da manipulação de cópia, é importante referir que a atribuição das permissões é realizada nos componentes gráficos da atividade que permitem seleção. Quando o profissional de saúde tenta selecionar a informação para a copiar, o Android responde de modo diferente consoante a permissão atribuída. Se permitir a seleção, esta responde com as ações para copiar e colar (Ilustração 16), caso contrário, nada aparece.

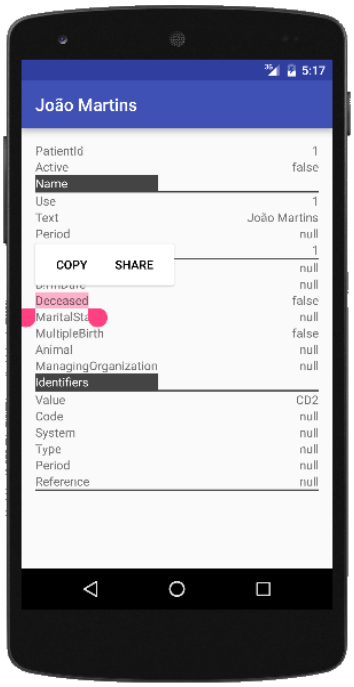



**Ilustração 16 - Operação de manipulação copia (*copy paste*)**

Na execução da manipulação de captura de ecrã, o Android irá sempre responder. Consoante a permissão atribuída ao profissional de saúde, a resposta poderá ser diferente. E de notar que essa permissão é atribuída a cada atividade da Aplicação de Visualização.

Quando o profissional de saúde tentar uma captura de ecrã e tem permissão, a captura de ecrã resulta como indicado na Ilustração 17. Quando este não tem permissão, o resultado consiste numa captura de ecrã em que o resultado não apresenta qualquer informação, como indicado na Ilustração 18.

É importante referir que quando o profissional de saúde entra numa qualquer atividade, a Aplicação de Visualização volta a confirmar as permissões de manipulação definidas.

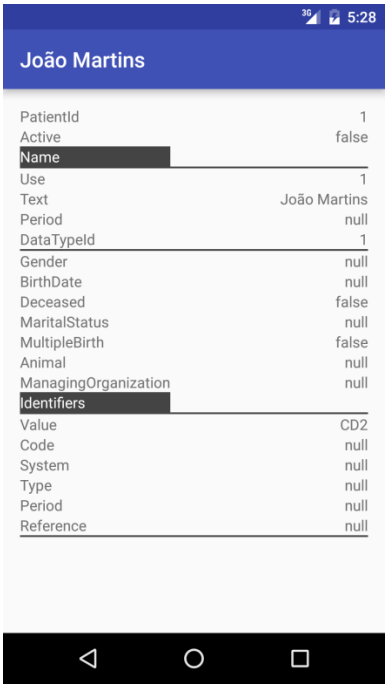



**Ilustração 17 - Operação de manipulação captura de ecrã com sucesso**

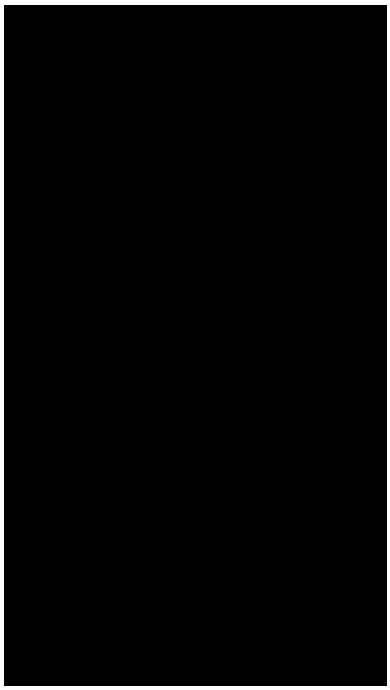


**Ilustração 18 - Operação de manipulação captura de ecrã sem sucesso**

## 4.4 Avaliação

Esta secção irá apresentar uma avaliação da implementação. Esta terá como base os efeitos observados na execução da demostração e da implementação realizada, quando comparados objetivos definidos. A razão de ser necessário utilizar a implementação para a avaliação, tem a ver com o facto de alguns objetivos necessitarem de uma implementação que não irá refletir-se na demostração ao utilizador. O objetivo de conseguir partilhar registos clínicos em formatos utilizados em serviços web consiste num desses objetivos.

A avaliação desse objetivo é positiva, isto porque é possível verificar na implementação que todos os recursos disponibilizados pelos servidores são realizados utilizando o serviço web RESTful. A aplicação de visualização utiliza uma biblioteca presente no Android para requisitar tais recursos disponibilizados.

Outro objetivo que também só poderá ser avaliado com a implementação consiste em conseguir registar todas as ações realizadas ou tentadas sobre os registos clínicos. Utilizando a camada Business, os servidores registam na tabela da base de dados, todos os pedidos realizados. Isto inclui a origem e resultado. Dependo do tipo de pedido, a informação poderá ser guardada em diferentes tabelas.

Um dos objetivos que envolve a Aplicação de Visualização consiste em implementar ou utilizar um mecanismo que permita que aplicações não-autorizadas não tenham acesso direto aos registos clínicos disponibilizados. A avaliação deste objetivo terá ser visto em duas vertentes. No armazenamento no dispositivo ou na memória da aplicação. O mecanismo para controlar a informação armazenada, tem como base a encriptação. Utilizando a chave adquirida no Servidor de Permissões, a informação e encriptada e depois armazenada. Na memória da aplicação, a situação é diferente, isto porque esta não está encriptada devido à necessidade de mostrar ao utilizador. Cabe ao sistema operativo garantir que ninguém consegue aceder diretamente. No caso da Aplicação de Visualização, cabe a esta garantir que a sua memória é limpa assim que a aplicação é destruída, de modo a garantir que pedaços da informação clínica não ficam disponíveis para outras aplicações explorarem.

Por fim, o último objetivo consiste em permitir controlar as ações que um utilizador pode realizar quando está a aceder a informação clinica, tais como, visualização, impressão, cópia, captura de ecrã, entre outras. Este objetivo é o único que é visível para o utilizador final. Para conseguir tal controlo, foram utilizados os métodos que os componentes gráficos dispõem para controlar as operações de manipulação. As autorizações de manipulação foram adquiridas através de um pedido ao servidor de permissões.

# Capítulo 5

# Conclusão

Esta dissertação teve como fim perceber como as tecnologias, usadas para controlar e monitorizar informação partilhada (IRM), poderiam ser adaptadas de modo a realizar as mesmas funções, mas para informação clínica partilhada via RESTful. A razão para tal tem a ver com o facto de estas tecnologia apenas funcionarem com informação partilhada em formato de documento. Esta limitação levou às seguintes questões de investigação:

- Será possível implementar um mecanismo, tendo como base as ideias das tecnologias IRM, que permita controlar e monitorizar informação clínica partilhada via serviços web RESTful?
- Quais as limitações que esse mecanismo terá?

Para alinhar a primeira questão de investigação com esta dissertação, foram definidos um conjunto de objetivos no capítulo dos Objetivos. Esses objetivos foram tidos em conta na definição da solução ideal, assim como, na implementação. Após uma análise da implementação e demostração, conclui-se que é possível implementar um mecanismo que consegue monitorizar informação clínica partilhada via serviços web RESTful. As principais razões para tal afirmação têm a ver com dois factos: a existência de uma aplicação que consegue comunicar com servidores que utilizam serviços web RESTful de modo a adquirir informação clínica ou outros tipos de informação; e a capacidade dessa aplicação de encapsular, dentro das possibilidades, a informação recebida, implementando mecanismos de controlo monitorização sobre esta.

Já na segunda questão de investigação, a principal limitação tem a ver com o desenvolvimento da aplicação responsável por garantir o cumprimento das políticas de segurança. Dependendo da tecnologia utilizada para desenvolver tal aplicação, esta poderá dificultar a implementação de mecanismos mais granulares de controlo. No caso da implementação apresentada, foi utilizado a linguagem de programação Java em ambiente Android. Esta é uma linguagem de alto nível. Isto significa que existe uma camada de abstração entre o programador e o código máquina. Esta camada de abstração faz com o programador não se tenha de preocupar com algumas realidades, tais como, a alocação e limpeza de memória. No entanto, ao desenvolver aplicações que usam informação secreta, como por exemplo chaves de encriptação, é importante ter esse controlo, de modo a obter maior segurança e evitar que esta fique em memória após a aplicação ter terminado.

Infelizmente uma limitação do trabalho realizado prende-se com a validação do mesmo. Não foi possível validar a arquitetura e o sistema desenvolvido, recorrendo à sua utilização por parte de utilizadores, e depois avaliar a opinião dos mesmos na utilização do sistema. Por outro lado, não foi possível realizar testes de segurança por parte de especialistas que pudessem atestar a segurança do sistema desenvolvido. As conclusões apresentadas resultam do grau de cumprimento dos objetivos estabelecidos para esta trabalho assim como da resposta às diversas questões de investigação.

Para consultar a implementação apresentada nesta dissertação, é disponibilizado o seguinte *link*:

- *https://drive.google.com/drive/folders/0BxX63H4CO7UMS3VmbkxmcW9xaGM?usp=sharing*

## 5.1 Trabalho Futuro

A solução apresentada permitiu concluir que é possível implementar um mecanismo para controlar e monitorizar informação clínica partilhada, no entanto, é ainda necessária uma investigação mais profunda relativo ao sistema operativo.

O sistema operativo é responsável, até certo ponto, por garantir que cada aplicação utiliza apenas o seu espaço de memória. Para tal, utiliza diversos tipos de mecanismo. Uma investigação mais profunda sobre quais e como esses mecanismos funcionam, poderá ajudar no desenvolvimento mais seguro de aplicações responsáveis por garantir o cumprimento das políticas de segurança, no sentido de obter mais informações sobre as limitações existentes. Essa investigação não deveria focar-se apenas no sistema operativo Android. Deverá abranger uma panóplia dos sistemas operativos mais utilizados, tais como, iOS, Mac e Windows. Seria importante poder comparar os mecanismos investigados, de modo a perceber até que ponto seria possível assegurar o isolamento do espaço de memória utilizado por cada aplicação.

# Referências

Alex Rodriguez. (9 de fevereiro de 2015). *RESTful Web services: The basics.* Obtido em 9 de janeiro de 2016, de IBM Developer Work: http://www.ibm.com/developerworks/webservices/library/ws-restful

Alonso, G., Casati, F., Kuno, H., & Machiraju, V. (2004). Web Services. *Web Services - Concepts, Architectures and Applications, Chapter 1*, 397-405.

Android Developers. (2016). *Activities*. Obtido em 10 de September de 2016, de Android Developers: https://developer.android.com/guide/components/activities.html

California Health Care Foundation. (março de 2014). *ELINCS: The National Lab Data Standard for Electronic Health Records.* Obtido em 19 de fevereiro de 2016, de California Health Care Foundation: http://www.chcf.org/projects/2009/elincs

Çetinkaya, H. B. (25 de maio de 2014). *REST & RESTful Web Services.* Obtido em 10 de janeiro de 2016, de Slideshare: http://pt.slideshare.net/hburakcetinkaya/rest-res-tful-web-services

Corepoint Health. (2010). *The HL7 Evolution.* Obtido em 26 de janeiro de 2016, de Corepoint Health: http://corepointhealth.com/whitepapers/evolution-hl7

Curbera, F., Duftler, M., Khalaf, R., & Nagy, W. (2002). Unraveling the Web Services Web An Introduction to SOAP, WSDL,and UDDI. *Ieee Internet Computing*, 86-93.

Dreyer, K. J. (2000). Why IHE? *Radiographics*, 1583-1584.

E. Hammer-Lahav. (abril de 2010). *The OAuth 1.0 Protocol.* Obtido em 5 de fevereiro de 2016, de Internet Engineering Task Force: http://tools.ietf.org/html/rfc5849

Entity Framework Tutorial. (2016). *What is Entity Framework?* Obtido em 9 de September de 2016, de Entity Framework Tutorial: http://www.entityframeworktutorial.net/what-is-entityframework.aspx

FHIR. (24 de outubro de 2015). *FHIR Overview.* Obtido em 31 de janeiro de 2016, de HL7: https://www.hl7.org/fhir/overview.html

*FHIR Security.* (24 de outubro de 2015). Obtido em 3 de fevereiro de 2016, de HL7 FHIR: https://www.hl7.org/fhir/security.html

FMUP. (2006). *Registos clínicos.* Obtido em 30 de janeiro de 2016, de Faculdade de medicina da universidade do Porto: http://im.med.up.pt/epr/epr.html

Fredrich, T. (2012). RESTful Service Best Practices Recommendations for Creating Web Services. 1-25.

Geoff Anderson. (2008). What is: Information Rights Management? *Management*, 2003-2004.

HIMSS. (17 de agosto de 2015). *Integrating HL7 into Medical Applications with LEADTOOLS.* Obtido em 20 de fevereiro de 2016, de Healthcare Information and Management Systems Society: http://www.himss.org/ResourceLibrary/genResourceDetailPDF.aspx?ItemNumber=43775

HL7. (24 de outubro de 2015). *Introducing HL7 FHIR.* Obtido em 25 de janeiro de 2016, de HL7: http://hl7.org/fhir/summary.html

HL7. (2016). *HL7 Standards - Section 4: EHR Profiles.* Obtido em 20 de fevereiro de 2016, de Health Level Seven: http://www.hl7.org/implement/standards/product_section.cfm?section=4

HL7. (2016). *HL7 Version 3 Standard: Transport Specifications - MLLP.* Obtido em 25 de janeiro de 2016, de HL7: http://www.hl7.org/implement/standards/product_brief.cfm?product_id=55

HL7. (2016). *Introduction to HL7 Standards.* Obtido em 23 de janeiro de 2016, de HL7: http://www.hl7.org/implement/standards/index.cfm?ref=nav

Howe, D. (17 de fevereiro de 1999). *Dictionary.* Obtido em 25 de janeiro de 2016, de Lower layer protocol: http://dictionary.reference.com/browse/lower-layer-protocol

Hunsaker, C. (18 de maio de 2015). *REST vs SOAP: When Is REST Better?* Obtido em 20 de fevereiro de 2016, de https://stormpath.com/blog/rest-vs-soap/

*Information Rights Management*. (23 de July de 2015). Obtido em 13 de September de 2016, de Microsoft: https://technet.microsoft.com/en-us/library/dd638140(v=exchg.150).aspx#Anchor_1

Interfaceware. (2016). *Iguana.* Obtido em 31 de janeiro de 2016, de Interfaceware: http://www.interfaceware.com/iguana.html

LinkedCare. (2016). *Ligação Médico-Paciente.* Obtido em 6 de fevereiro de 2016, de LinkedCare: http://www.linkedcare.com/share.html

Mayko, L. (12 de junho de 2015). *Choosing an OAuth Type for Your API* . Obtido em 5 de fevereiro de 2016, de API2Cart: https://www.api2cart.com/blog/choosing-oauth-type-api/

Merenyi, R. (18 de março de 2013). *What Are Web Services and Where Are They used?* Obtido em 9 de janeiro de 2016, de Segue Technologies: http://www.seguetech.com/blog/2013/03/18/where-web-services-used

Michael McFarland, S. (junho de 2012). *Why We Care about Privacy?* Obtido em 10 de janeiro de 2016, de Santa clara university: http://www.scu.edu/ethics/practicing/focusareas/technology/internet/privacy/why-care-about-privacy.html

Microsoft. (2016). *ASP.NET Web API.* Obtido em 9 de September de 2016, de MSDN: https://msdn.microsoft.com/en-us/library/hh833994(v=vs.108).aspx

OpenEHR. (2016). *What is openEHR?* Obtido em 14 de September de 2016, de OpenEHR: http://www.openehr.org/what_is_openehr.php#

Oracle. (s.d.). *What is Oracle IRM?* Obtido em 2 de fevereiro de 2016, de Oracle: https://docs.oracle.com/cd/E29542_01/doc.1111/e12450/idteuwhatisoirm.htm#IDTEU110

Pal, A. K. (2014). Application of digital rights management in library. *DESIDOC Journal of Library and Information Technology*, 11-15.

Pautasso, C., Zimmermann, O., & Leymann, F. (2008). Restful web services vs. 'big'web services: making the right architectural decision. *Proceeding of the 17th international conference on World Wide Web*, 805–814.

Perrin, C. (15 de agosto de 2007). *The three elements of access control.* Obtido em 29 de janeiro de 2016, de Tech Republic: http://www.techrepublic.com/blog/it-security/the-three-elements-of-access-control/

Pingdom. (15 de outubro de 2010). *REST in peace, SOAP.* Obtido em 26 de dezembro de 2015, de pingdom: http://royal.pingdom.com/2010/10/15/rest-in-peace-soap/

Serviços Partilhados do Ministério da Saúde. (2014). Obtido em 30 de janeiro de 2016, de Portal do Utente: https://servicos.min-saude.pt/utente/

Steven Davelaar. (27 de fevereiro de 2015). *Performance Study – REST vs SOAP for Mobile Applications.* Obtido em 20 de fevereiro de 2016, de Oracle A-Team: http://www.ateam-oracle.com/performance-study-rest-vs-soap-for-mobile-applications/

Subramanya, S., & Yi, B. (2006). Digital rights management. *IEEE Potentials*, 31-34.

Tutorials Point. (2016). *MVC Framework - Introduction*. Obtido em 9 de September de 2016, de Tutorials Point: http://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm

Zhang, X. (28 de novembro de 2011 ). *A Survey of Digital Rights Management Technologies.* Obtido em 8 de janeiro de 2016, de Washington University in St. Louis: http://www.cse.wustl.edu/~jain/cse571-11/ftp/drm/

Zhao, H., & Doshi, P. (2009). Towards automated restful web service composition. *IEEE International Conference on Web Services*, 189-196.