

Universidade do Minho
Mestrado Integrado em Engenharia Informática
4^o ano - 2^o Semestre

Interoperabilidade Semântica

Folha de Exercício FE01

Grupo 01



A83732 – Gonçalo Rodrigues Pinto

26 de Março de 2021

Conteúdo

1	Introdução	3
2	Descrição do Problema	3
3	Descrição do Trabalho	4
3.1	Tecnologias utilizadas	4
3.2	Caracterização das Bases de Dados	4
3.3	Interface Web	5
3.3.1	Aplicação A	6
3.3.2	Aplicação B	9
3.4	Construção das Mensagens HL7	10
3.5	Emissão e recepção de mensagens	11
3.5.1	Aplicação A para Aplicação B	12
3.5.2	Aplicação B para Aplicação A	13
4	Resultados	14
5	Conclusão	15

Lista de Figuras

1	Base de Dados implementada.	5
2	Operações possíveis de executar na aplicação A.	6
3	Registar novo episódio.	6
4	Registar novo tipo de exame.	6
5	Registar novo paciente.	7
6	Registar novo pedido.	7
7	Visualizar a lista de trabalho (ver estado dos pedidos e possibilidade de cancelar o estado dos mesmos).	7
8	Informação sobre um determinado exame (visualização do relatório).	8
9	Informação sobre um determinado paciente.	8
10	Informação sobre um determinado episódio.	8
11	Visualização da lista de episódios.	8
12	Operações possíveis de executar na aplicação B.	9
13	Escrever e publicar o relatório.	9
14	Ver estado do pedidos na aplicação B.	9
15	Ver pedidos por realizar na aplicação B (cancelar ou finalizar pedido.	9
16	Ver pedidos cancelados na aplicação B	10
17	Ver pedidos finalizados na aplicação B.	10
18	Mensagem recebida pela aplicação B enviada pela aplicação A.	12
19	ACK recebido pela aplicação A após enviar a mensagem para B.	12
20	Mensagem recebida pela aplicação A enviada pela aplicação B.	13
21	ACK recebido pela aplicação B após enviar a mensagem para A.	13

1 Introdução

No 2º semestre do 1º ano do Mestrado em Engenharia Informática da Universidade do Minho, existe uma unidade curricular enquadrada no perfil Engenharia do Conhecimento denominada por Interoperabilidade Semântica, que tem como objetivo a introdução ao conceito de interoperabilidade semântica.

No setor da saúde, a interoperabilidade é a capacidade que os diversos sistemas da informação e aplicativos de software têm de comunicar, trocar dados e utilizar as informações trocadas. A interoperabilidade tem um valor importante no aumento da segurança do paciente ao permitir o acesso e a disponibilidade aos dados clínicos dos pacientes. O acesso aos dados clínicos do paciente em tempo real, permite ao sistema de saúde atender um paciente a partir de qualquer local de atendimento do sistema melhorando, desta forma, a qualidade e a eficiência desse atendimento. É imperativo que os diversos sistemas de saúde possam trocar informações e transferi-las de um sistema para outro por meio de interfaces específicas, adaptadas ou personalizadas, que estructurem as informações de maneira semelhante. Para conseguir a troca de informações da maneira mais natural possível, é imprescindível a adoção de padrões sobre os quais, os diferentes sistemas de saúde possam coincidir. O setor da saúde desenvolveu-se e adotou padrões para vários propósitos relacionados com mensagens de texto, terminologia, documentos, mapas conceituais, aplicações e arquiteturas, foram desenvolvidos padrões que definem o formato e a estrutura de elementos de dados para facilitar a comunicação entre diversos sistemas clínicos, um desses padrões é o padrão HL7 para trocar dados demográficos, clínicos e administrativos.

O presente trabalho enquadra-se na unidade curricular de Interoperabilidade Semântica e pretende que os alunos desenvolvam um sistema para envio e receção de mensagens HL7.

2 Descrição do Problema

Como foi referido previamente o objetivo deste trabalho passou pelo desenvolvimento de um sistema para envio e receção de mensagens HL7, para esse efeito foram desenvolvidas duas aplicações.

Numa delas pretendeu-se criar uma base de dados e desenvolver um pequeno programa para registar pedidos de exames médicos. Um pedido deverá ter um número, uma data e hora em que foi registado, um doente (identificado por um número sequencial, um número de processo, uma morada, um telefone e um género). O pedido deverá ser feito num contexto de um episódio (identificado por um número de episódio) em contexto de Consulta (CON). O pedido refere-se a um ato médico, sendo também acrescentada alguma informação clínica. Sempre que haja um novo pedido ou uma alteração ao pedido (incluindo um cancelamento) deverá ser criado um registo numa lista de trabalho (tabela da base de dados).

Na outra aplicação criada criou-se outra base de dados e foi desenvolvido um programa onde serão realizados os exames pedidos na primeira máquina e produzido o respetivo relatório em formato texto. Os exames serão carregados na base de dados a partir de uma lista de trabalho (tabela da segunda base de dados). Sempre que um exame seja realizado e um relatório produzido deverá atualizar a primeira base de dados com um novo estado e com o relatório, que poderá ser visualizado.

Posteriormente foi desenvolvido um cliente para enviar mensagens HL7 e um servidor para receber mensagens HL7 nas duas aplicações.

Qualquer partilha de informação entre as duas bases de dados apenas foi feita através do recurso à norma HL7.

3 Descrição do Trabalho

Para o desenvolvimento de um sistema para envio e receção de mensagens HL7 foi necessário efetuar um levantamento de características e estudo das mesmas, com o objetivo de perceber as melhores tecnologias bem como a melhor estrutura aplicacional para a aplicação a desenvolver.

3.1 Tecnologias utilizadas

A apresentação das diversas informações ao utilizador requer uma componente capaz de ler conteúdos da Base de Dados, interpretando-os para a interface gráfica. Para este efeito, foi decidido criar uma interface Web baseada em **Pug** [10] onde a *framework* **Express** [1] disponibilizada pelo **Node.js** [9] (a linguagem utilizada foi **Javascript**) fornece os dados e funcionalidades de gestão de uma forma apelativa, intuitiva e de fácil utilização para isso a informação foi armazenada na base de dados relacional **MySQL** [8] pois esta permite a separação dos diferentes conceitos.

Para o apoio na criação das mensagens HL7 e comunicação entre cliente e servidor das duas aplicações foi utilizado a *framework* **Simple-hl7** [11] porque é uma biblioteca simples para a criação de *middleware* HL7, baseada em *Connect & Express* tornando as interfaces e *middleware* HL7 fáceis de utilizar quanto os servidores Web (por exemplo, Mirth). Esta biblioteca permite fazer o *Parsing* das mensagens HL7, facilita uma API de construção e edição de mensagens HL7 e fornece componentes de servidor/cliente TCP, sendo esta a razão principal pela escolha desta *framework*.

3.2 Caracterização das Bases de Dados

Ambas as aplicações desenvolvidas partilham dados, assim ambos os sistemas de registo de pedidos e realização de exames precisam de armazenar a mesma informação, por isso ambas as bases de dados são iguais, mas poderiam muito bem ter uma estrutura diferente que permitisse guardar a mesma informação.

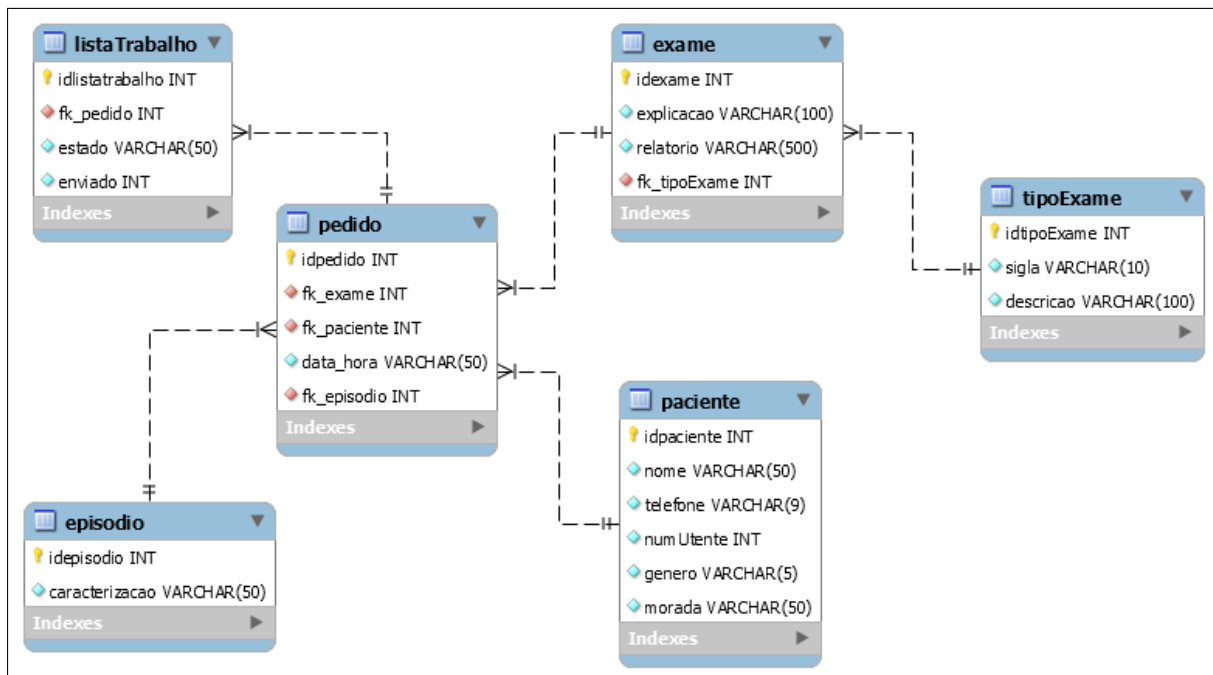


Figura 1: Base de Dados implementada.

Como pode ser observado na figura 1 foram consideradas as seguintes entidades (tabelas) com os seguintes atributos (colunas):

- **Tipo de Exame** referente ao contexto do pedido, possui um identificador, uma sigla e uma descrição do tipo de exame em causa;
- **Exame** descreve o exame a realizar através de um identificador único, a explicação que justifique o exame, o respetivo relatório e o contexto do tipo de exame em causa, através de uma chave estrangeira referente à entidade Tipo de Exame;
- **Paciente** descreve o doente identificado por um número sequencial, um número de processo, uma morada, um telefone e género;
- **Episódio** contextualiza um determinado pedido que foi feito ou irá ser feito;
- **Pedido** unidade fulcral da aplicação possui um identificador único e referência ao episódio, exame e paciente (através de chaves estrangeiras);
- **Lista de Trabalho** permite o registo de um novo pedido ou uma alteração do estado do pedido, para isso possui a identificação do pedido em causa, o estado deste e um campo a representar que a informação já foi comunicada à outra aplicação.

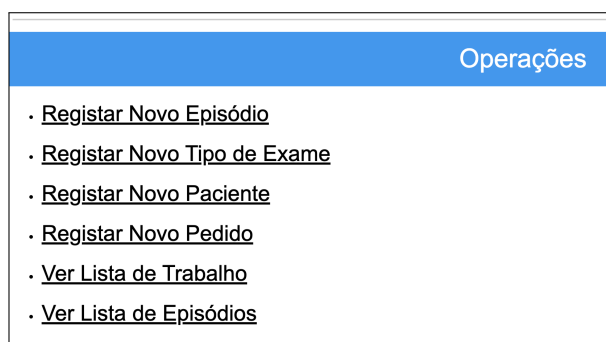
3.3 Interface Web

Como foi descrito previamente foi construído uma aplicação Web, baseada em **Pug** com a *framework* **Express** disponibilizada pelo **Node.js**, para cada uma das aplicações de forma a ser possível efetuar as operações que permitem o registo de um novo pedido, alteração do estado do pedido e produção do respetivo relatório em formato texto.

3.3.1 Aplicação A

Nesta aplicação foram desenvolvidas as seguintes operações:

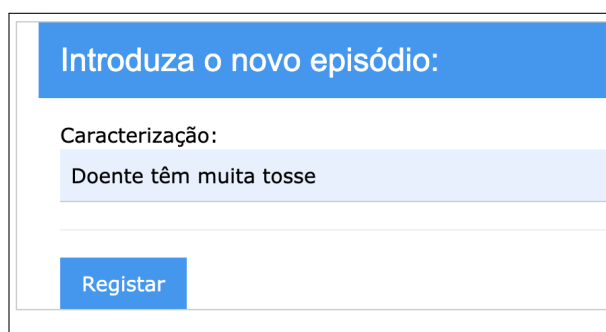
- Registrar novo Episódio, novo Tipo de Exame, novo Paciente e novo Pedido;
- Ver o estado do pedido;
- Cancelar o pedido;
- Ver o relatório.



Operações

- Registrar Novo Episódio
- Registrar Novo Tipo de Exame
- Registrar Novo Paciente
- Registrar Novo Pedido
- Ver Lista de Trabalho
- Ver Lista de Episódios

Figura 2: Operações possíveis de executar na aplicação A.



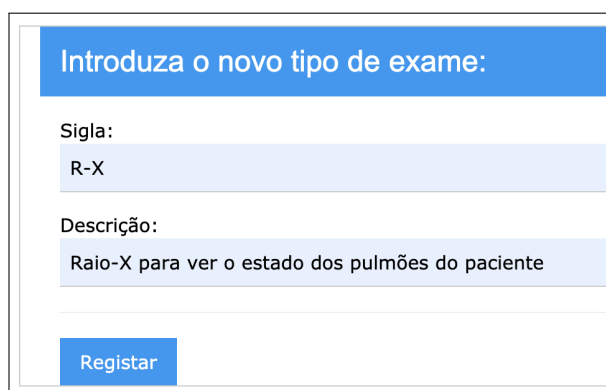
Introduza o novo episódio:

Caracterização:

Doente têm muita tosse

Registrar

Figura 3: Registrar novo episódio.



Introduza o novo tipo de exame:

Sigla:

R-X

Descrição:

Raio-X para ver o estado dos pulmões do paciente

Registrar

Figura 4: Registrar novo tipo de exame.

Introduza o novo paciente:

Nome:

José António

Telefone:

911991199

Número Utente:

123456789

Morada:

Rua exemplo nº49 São Vitor, Braga

Género:

Masculino

Registrar

Figura 5: Registrar novo paciente.

Introduza o novo pedido:

Tipo de Exame:

Raio-X para ver o estado dos pulmões do paciente (R-X)

Descrição :

Necessito de ver o estado dos pulmões do paciente.

Paciente :

José António : 123456789

Data :

16/03/2021, 17:30

Episódio :

2 : Doente têm muita tosse

Registrar

Figura 6: Registrar novo pedido.

Lista de Trabalho						
ID_Pedido	ID_Exame	ID_Paciente	Data_Hora	Episódio	Estado	Cancelar Pedido
4	4	2	2021-03-16T17:30	2	Por realizar	X

Figura 7: Visualizar a lista de trabalho (ver estado dos pedidos e possibilidade de cancelar o estado dos mesmos).

Em cada linha da lista de trabalho é possível carregar no identificador do exame, do paciente ou do episódio levando o utilizador a visualizar mais informações sobre cada uma destas entidades.

Exame: 4					
Descrição	Necessito de ver o estado dos pulmões do paciente.				
Relatório	desconhecido				
Tipo de Exame	<table> <tr> <td>Sigla</td><td>R-X</td></tr> <tr> <td>Descrição</td><td>Raio-X para ver o estado dos pulmões do paciente</td></tr> </table>	Sigla	R-X	Descrição	Raio-X para ver o estado dos pulmões do paciente
Sigla	R-X				
Descrição	Raio-X para ver o estado dos pulmões do paciente				

Figura 8: Informação sobre um determinado exame (visualização do relatório).

Paciente: 123456789	
Nome	José António
Telefone	911991199
Número Utente	123456789
Morada	Rua exemplo nº49 São Vitor, Braga
Género	M

Figura 9: Informação sobre um determinado paciente.


Episódio: 2				
Caracterização	Doente têm muita tosse			
Pedidos Existentes	ID_Pedido	Data_Hora	Estado	Ver exame
	4	2021-03-16T17:30	Por realizar	

Figura 10: Informação sobre um determinado episódio.

Lista de Episódios		
ID_Episódio	Caracterização	Total Pedidos
2	Doente têm muita tosse	1
1	Falta de ar	3

Figura 11: Visualização da lista de episódios.

Em cada linha da lista de episódios é possível obter mais informação através do identificador de cada episódio, onde nesta é possível observar todos os pedidos referentes ao episódio em causa e posteriormente visualizar o exame em específico.

3.3.2 Aplicação B

Nesta aplicação foram desenvolvidas as seguintes operações:

- Ver estado do pedido;
- Cancelar pedido;
- Finalizar o pedido;
- Escrever e publicar o relatório.

Operações
<ul style="list-style-type: none">• <u>Escrever e publicar relatório</u>• <u>Ver estado dos pedidos</u>

Figura 12: Operações possíveis de executar na aplicação B.

Introduza o novo relatório:
Exame: (R-X) 1615915800000 - :: José António_123456789 :: Ep2 ▾
Relatório:
<div>O paciente têm muita expectoração nos pulmões.</div>
<div>Publicar</div>

Figura 13: Escrever e publicar o relatório.

Ver estado dos pedidos
<ul style="list-style-type: none">• <u>Por Realizar</u>• <u>Cancelados</u>• <u>Finalizados</u>

Figura 14: Ver estado do pedidos na aplicação B.

Pedidos Por_Realizar							
ID_Pedido	ID_Exame	ID_Paciente	Data_Hora	Episódio	Estado	Cancelar Pedido	Finalizar Pedido
4	4	2	1615915800000	2	Por realizar	X	V

Figura 15: Ver pedidos por realizar na aplicação B (cancelar ou finalizar pedido).

Pedidos Cancelados							
ID_Pedido	ID_Exame	ID_Paciente	Data_Hora	Episódio	Estado	Cancelar Pedido	Finalizar Pedido
3	3	1	1616095920000	1	Cancelado		

Figura 16: Ver pedidos cancelados na aplicação B

Pedidos Finalizados							
ID_Pedido	ID_Exame	ID_Paciente	Data_Hora	Episódio	Estado	Cancelar Pedido	Finalizar Pedido
2	2	1	1615317780000	1	Finalizado		
1	1	1	NaN	1	Finalizado		

Figura 17: Ver pedidos finalizados na aplicação B.

Em cada linha da tabela dos diferentes estados é possível carregar no identificador do exame, do paciente ou do episódio levando o utilizador a visualizar mais informações sobre cada uma destas entidades.

3.4 Construção das Mensagens HL7

Tal como foi referido para o apoio na criação das mensagens HL7 foi utilizado a *framework* **Simple-hl7**, esta facilita uma API de construção e edição de mensagens HL7.

Uma mensagem HL7 utiliza uma sintaxe de codificação legível por humanos, baseada em segmentos (linhas) e delimitadores de um caractere. Os segmentos têm componentes (campos) separados por um delimitador de componentes. Um componente pode ter mais subcomponentes, que, por sua vez, são separados pelo delimitador de subcomponentes. Os delimitadores padrões são a barra vertical, para o separador de campo, o acento circunflexo para o separador de componentes, e comercial para o separador de subcomponentes. O til é o separador de repetição padrão. Cada segmento começa com uma cadeia de 3 caracteres, que identifica o tipo de segmento. Cada segmento da mensagem contém uma categoria específica de informações. Cada mensagem tem um MSH como o seu primeiro segmento, que inclui um campo que identifica o tipo de mensagem. O tipo de mensagem determina os tipos de segmentos esperados na mensagem. Os tipos de segmentos usados num determinado tipo de mensagem são especificados pela notação gramatical de segmento utilizado nas normas HL7.

De forma a perceber os segmentos que permitem armazenar a informação devida foi necessário recorrer a documentação disponível, assim sendo, considerou-se que os segmentos fulcrais para enviar a informação armazenada na base de dados foram:

1. **MSH** (Message Header)[6] este segmento é obrigatório, sendo responsável por transmitir o identificador do pedido em causa;
2. **PID** (Patient Identification)[3] permite trocar informação acerca da identidade do paciente;
3. **PV1** (Patient Visit)[5] possibilita conhecer o episódio ao qual este pedido se realiza;
4. **ORC** (Common Order)[7] este segmento proporciona conhecer o estado do pedido;
5. **OBX** (Observation Request)[2] usado para transmitir informações específicas para um pedido de estudo diagnóstico ou observação, exame físico ou avaliação;
6. **OBX** (Observation/Result)[4] é usado para transmitir uma única observação ou fragmento de observação. Representa a menor unidade indivisível de um relatório. O segmento OBX também pode conter dados encapsulados. A principal missão é levar informações sobre observações em mensagens de relatórios.

Cada segmento possui um tamanho fixo de campos onde cada um representa uma determinada informação, mais uma vez foi fundamental recorrer à documentação disponibilizada pois permitiu colocar os dados presentes no campo correto.

3.5 Emissão e receção de mensagens

A comunicação entre cliente e servidor das duas aplicações foi realizada com a *framework* **Simple-hl7** porque esta fornece componentes de servidor/cliente TCP.

Assim, foi criado um servidor TCP em cada uma das aplicações construídas a partir da *framework* utilizada, este servidor encontra-se à escuta numa determinada porta definida. O servidor na aplicação A é responsável por atualizar o estado de um determinado pedido e também atualizar o relatório de um pedido. Por outro lado, o servidor da aplicação B regista novos pedidos e se necessário novas informações, além de que, é responsável por atualizar o estado dos pedidos.

Para os servidores executarem as operações acima mencionadas foi necessário criar clientes em cada uma destas aplicações, onde o cliente na aplicação A envia a informação que ainda não foi enviada para aplicação B, podendo esta ser relativa a um novo pedido ou alteração do estado deste. Já o cliente da aplicação B é encarregue de informar a aplicação A que houve alterações no estado dos pedidos e/ou adição de relatórios.

3.5.1 Aplicação A para Aplicação B

```
*****message received*****
MSH|^~\&|AppA|AppA|AppB|AppB|1615915652345||ORM|4|P|2.5|4|||PT|PT|
PID|4|2||José António||M||Rua exemplo nº49 São Vitor, Braga|PT|911991199|PT||123456789|||PT|PT|
PV1|I|R-X|||||Doente têm muita tosse|||||2|
ORC|NW|4727374|4727374|HD||1615915652347|AppB|AppB||1615915652347||AppA|
OBR|2|4727374|4727374||1615915800000||||Necessito de ver o estado dos pulmões do paciente.
||Raio-X para ver o estado dos pulmões do paciente|||
*****sending ack*****
```

Figura 18: Mensagem recebida pela aplicação B enviada pela aplicação A.

```
Sending... message!
MSH|^~\&|AppA|AppB|AppA|20210316172732||ACK|ACK20210316172732|P|2.3
MSA|AA|ORM
```

Figura 19: ACK recebido pela aplicação A após enviar a mensagem para B.

De forma a que a aplicação B conheça a informação presente na aplicação A, o cliente da aplicação A consulta de 10 em 10 segundos a lista de trabalho desta aplicação verificando quais os pedidos que ainda não foram enviados, para isso é consultado o atributo "enviado" (se encontrar o valor a 0 significa que a informação ainda não foi comunicada), para cada pedido é efetuado um conjunto de verificações no sentido de transformar o estado atual do pedido na sintaxe HL7 (por exemplo, para representar que o pedido encontra-se cancelado, é representado como "CA"), posteriormente é construído a mensagem HL7 com os segmentos devidos (neste caso, são necessários o MSH, PID, PV1, ORC e OBR) com a informação do pedido e finalmente é enviado ao servidor da aplicação B. Como foi apresentado na aplicação B esta vai registar os pedidos recebidos; assim numa primeira fase é consultado o campo 5 do segmento ORC no sentido de descodificar o estado do pedido recebido; de seguida, é consultada a lista de trabalho da aplicação B no sentido de verificar que o identificador do pedido recebido já existe, se existir é atualizado o estado do pedido, caso contrário é verificada a informação das diversas tabelas que dependem através dos identificadores recebidos na mensagem, se não existir alguma informação, esta é inserida acedendo aos dados do segmento correspondente, por outro lado se existir apenas se insere na lista de trabalho pois já se garantiu que os identificadores são os devidos.

3.5.2 Aplicação B para Aplicação A

```
*****message received*****
MSH|^~\&|AppB|AppB|AppA|AppA|1615918661476||ORM|4|P|2.5|4|||PT|PT|
PID|4|2||José António||M||Rua exemplo nº49 São Vitor, Braga|PT|911991199||PT||123456789|||PT|PT|
ORC|NW|4727374|4727374||HD||1615918661477|AppB|AppB||1615918661477|||AppA|
OBR|2|4727374|4727374||NaN||||Necessito de ver o estado dos pulmões do paciente.|||Raio-X
para ver o estado dos pulmões do paciente|||
OBX|1||Necessito de ver o estado dos pulmões do paciente.|||1615918661478|||AppB|Hospital|gr01
OBX|2||1615918661478|||AppB|Hospital|gr01
OBX|3||1615918661478|||AppB|Hospital|gr01
OBX|4||RELATORIO:|1615918661478|||AppB|Hospital|gr01
OBX|5||1615918661478|||AppB|Hospital|gr01
OBX|6||0 paciente têm muita expectoração nos pulmões.|||1615918661478|||AppB|Hospital|gr01
*****sending ack*****
```

Figura 20: Mensagem recebida pela aplicação A enviada pela aplicação B.

```
Sending... message!
MSH|^~\&|AppB|AppA|AppB|20210316181741||ACK|ACK20210316181741|P|2.3
MSA|AA|ORM
```

Figura 21: ACK recebido pela aplicação B após enviar a mensagem para A.

Para que a aplicação A conheça a informação presente na aplicação B, o cliente da aplicação B consulta de 10 em 10 segundos a lista de trabalho desta aplicação onde verifica quais os pedidos que ainda não foram enviados, para isso é consultado o atributo "enviado" (se encontrar o valor 0 significa que a informação ainda não foi comunicada), para cada pedido é efetuado um conjunto de verificações no sentido de transformar o estado atual do pedido na sintaxe HL7 (por exemplo, para representar que o pedido se encontra cancelado, é representado como "CA"). De seguida, é verificado o conteúdo do relatório, se este for desconhecido significa que houve uma alteração do estado do pedido é construída uma mensagem HL7 com os segmentos devidos (neste caso, são necessários o MSH, PID, PV1, ORC e OBR de forma a garantir que é atualizado a informação do pedido correto), caso contrário é criada uma mensagem HL7 com os seguintes segmentos MSH, PID, PV1, ORC, OBR e OBX. No segmento OBX o relatório é dividido em parágrafos, tal como é estabelecido os primeiros 5 segmentos representam informação variada e nos restantes é acrescentado cada parágrafo do relatório no campo *Observation Value*. O servidor da aplicação A após receber um pedido verifica se este contém segmentos OBX, se existir é necessário construir o relatório a partir do quinto segmento e posteriormente registar o relatório, como os identificadores existentes na aplicação B são os mesmos na aplicação A, a atualização é direta, é ainda verificado o estado do pedido recebido e caso seja diferente também é atualizado. Por outro lado, caso a mensagem não contenha segmentos OBX significa que o servidor da aplicação A apenas têm que atualizar o estado do pedido recebido.

4 Resultados

Em suma, após implementado uma interface Web para cada uma das aplicações, construção de mensagens HL7 e ainda a emissão e receção de mensagens neste formato foi possível colocar dois sistemas a interagir e a comunicar um com o outro.

Deste modo foi possível comparar as vantagens que a utilização de um formato como o HL7 com um formato/protocolo realizado de raiz, por um lado ao criamos nós um protocolo enviamos apenas a informação que achamos mais pertinente, contudo leva a que este seja apenas focado para um determinado caso de estudo, ao utilizar um formato robusto como HL7 é garantido que são cobertos grande parte dos problemas facilitando a interoperabilidade, mas é de realçar que a robustez traz também desvantagens, como por exemplo, a dificuldade de aprender HL7 que pode ser comparado como aprender o alfabeto após consolidado, é necessário aprender a ler e escrever, uma outra desvantagem prende-se com o facto de requer muito conhecimento sobre dados clínicos e processos de saúde.

De forma a contornar este segundo problema foi criado o *NextGen Connect* que é um mecanismo de interface multiplataforma ao qual foi recomendado a sua utilização neste trabalho. Este software é usado no setor de assistência médica e que permite a gestão de informações usando o envio bidirecional de muitos tipos de mensagens. O principal uso deste mecanismo de interface é na área da saúde, contudo tanto a nível individual como em grupo, não foi possível perceber como este software estabelece a comunicação entre as aplicações. Como foi recomendável instalou-se o *Mirth* com *docker* juntamente com *Mirth Connect Administrator*, após a instalação tanto o grupo como eu deparámos com alguns problemas, um desses passou pelo facto de não se conseguir ligar às bases de dados utilizadas e consequentemente não conseguir enviar informação possivelmente este problema pode ser explicado pela existência de duas instâncias *mysql* (uma no *docker*, outra localmente) contudo não conseguimos arranjar uma explicação lógica. De forma a solucionar este problema tentou-se como alternativa criar um cliente e servidor de forma a escrever as mensagens em HL7 para ficheiro, assim utilizar-se-ia apenas a comunicação entre canais que o *Mirth* disponibiliza, mas mais uma vez obtivemos sempre erros no *path* dos ficheiros.

Para isso solucionou-se através da utilização da *framework Simple-HL7* que facilitou na criação de mensagens só que em vez se escrever para um ficheiro, esta ferramenta disponibiliza a criação de servidor/cliente para tal foi necessário escrever o código de forma a efetuar as validações devidas e respetiva descodificação da informação recebida. No entanto, esta solução é pouco flexível pois apenas funciona neste formato caso se queira mudar a base da comunicação é necessário uma reestruturação no cliente e servidor de cada aplicação, ao contrário do *Mirth* que fornece e trabalha com diversos protocolos. Todavia, esperamos no futuro perceber o funcionamento de modo a poder utilizar em próximos trabalhos.

5 Conclusão

O presente relatório descreveu, de forma sucinta, o processo de construção de um protótipo para interoperação entre os dois sistemas enquadrado num determinado caso de estudo (pedido, realização e relato de exames médicos).

Durante a elaboração deste trabalho surgiram algumas dificuldades que, com o esforço e entusiasmo do grupo pelo resultado final, acabaram por ser ultrapassadas. Entre as quais destacam-se o desafio que foi descobrir quais os campos e respetivos segmentos que permitissem armazenar a informação a trocar, bem como, o envio e receção de mensagens HL7 no sentido de garantir que a informação não ficasse incompleta e consequentemente não existisse confusão ao nível das entidades, por exemplo, um relatório sobre um exame corresponder exatamente a um determinado pedido.

Numa perspetiva futura, considero que seria interessante a utilização do *Mirth Connect Integration Engine* no sentido de substituir a criação de cliente e servidor pois este sistema permite gerir informações usando um envio bidirecional de muitos tipos de mensagens.

Após a realização deste trabalho, fiquei consciente da importância que interoperabilidade entre sistemas de informação têm, esta é extremamente necessária no quotidiano pois ao existir um formato como o HL7 é possível desenvolver sistemas avançados que garantam que a informação não se perca e que não seja incoerente.

Considero que os objetivos propostos com a realização deste trabalho foram cumpridos, bem como, a consolidação dos conhecimentos na Interoperabilidade Semântica. Contudo tenho consciência que deveria ter investigado melhor o funcionamento da plataforma *Mirth Connect Integration Engine* de forma a poupar algum trabalho.

Por fim, espero que os conhecimentos obtidos e consolidados sejam de enorme utilidade tendo em conta uma perspetiva futura no mundo laboral.

Referências

- [1] *Express*. 2019. URL: <https://expressjs.com/> (acedido em 19/03/2021).
- [2] *HL7 v2.3 - OBR - Observation request segment*. 2021. URL: <https://hl7-definition.caristix.com/v2/HL7v2.3/Segments/OBR> (acedido em 19/03/2021).
- [3] *HL7 v2.4 - PID - Patient identification*. 2021. URL: <https://hl7-definition.caristix.com/v2/HL7v2.4/Segments/PID> (acedido em 19/03/2021).
- [4] *HL7 v2.5 - OBX - Observation/Result*. 2021. URL: <https://hl7-definition.caristix.com/v2/HL7v2.5/Segments/OBX> (acedido em 19/03/2021).
- [5] *HL7 v2.5 - PV1 - Patient Visit*. 2021. URL: <https://hl7-definition.caristix.com/v2/HL7v2.5/Segments/PV1> (acedido em 19/03/2021).
- [6] *HL7 v2.8 - MSH - Message Header*. 2021. URL: <https://hl7-definition.caristix.com/v2/HL7v2.8/Segments/MSH> (acedido em 19/03/2021).
- [7] *HL7 v2.8 - ORC - Common Order*. 2021. URL: <https://hl7-definition.caristix.com/v2/HL7v2.8/Segments/ORC> (acedido em 19/03/2021).
- [8] *mysql*. 2020. URL: <https://www.npmjs.com/package/mysql> (acedido em 19/03/2021).
- [9] *Node.js*. 2019. URL: <https://nodejs.org/en/> (acedido em 19/03/2021).
- [10] *Pug*. 2021. URL: <https://www.npmjs.com/package/pug> (acedido em 19/03/2021).
- [11] *simple-hl7*. 2020. URL: <https://www.npmjs.com/package/simple-hl7> (acedido em 19/03/2021).