Sistemas de Informação Distribuídos

Licenciaturas em Engenharia Informática e Informática e Gestão de Empresas

2019-2020, Segundo Semestre

Detecção de Intrusão e Incêndio em Museus

Mongo DB e Android

Identificação do grupo autor da especificação (Etapa A):

|  |  |  |
| --- | --- | --- |
| Número | Nome | Foto |
| 82493 | Miguel Diaz Gonçalves |  |
| 83380 | Gonçalo Dias do Amaral |  |
| 82361 | André Freitas |  |
| 82946 | Pedro Jones |  |
| 74278 | Dmytro Astashov |  |
| 73788 | Vitor Manuel Figueira Canhão |  |

Identificação do grupo autor da implementação (Etapas B):

|  |  |  |
| --- | --- | --- |
| Número | Nome | Foto |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

Índice

[1 Estrutura da Base de Dados Mongo 6](#_Toc38109446)

[2 Transporte Broker para Mongo (java) 7](#_Toc38109447)

[2.1 Periodicidade de Leitura de Sensores e Escrita no Mongo 7](#_Toc38109448)

[2.2 Utilização de Filtros que impeçam informação de ir para o Mongo 7](#_Toc38109449)

[3 Estrutura da Base de Dados Mysql 8](#_Toc38109450)

[3.1 Diagrama relacional 8](#_Toc38109451)

[3.2 Privilégios de Utilizadores 9](#_Toc38109452)

[4 Transporte Mongo para Mysql (java) 9](#_Toc38109453)

[4.1 Periodicidade de Leitura de Mongo e Escrita no Mysql 9](#_Toc38109454)

[4.2 Utilização de Filtros que impeçam informação de ir para o Mysql 9](#_Toc38109455)

[4.3 Mecanismo para assegurar que a migração é incremental 10](#_Toc38109456)

[5 Geração de Alertas 10](#_Toc38109457)

[5.1 Local/Momento de Deteção 10](#_Toc38109458)

[5.2 Regras para deteção de alertas 11](#_Toc38109459)

[Avaliação Global da Qualidade das Especificações 12](#_Toc38109460)

[6 Implementação 13](#_Toc38109461)

[6.1 Divergências face ao recebido/especificado 13](#_Toc38109462)

[6.1.1 Ideias que aproveitaram do outro grupo 13](#_Toc38109463)

[6.1.2 Ideias que abandonaram do próprio grupo 13](#_Toc38109464)

[6.1.3 Novas ideias introduzidas 13](#_Toc38109465)

[6.1.4 Código Mongo Implementado (dentro do java) 13](#_Toc38109466)

[7 Demonstração 13](#_Toc38109467)

[7.1 Situação 1 13](#_Toc38109468)

[7.1.1 Dados enviados pelo sensor 13](#_Toc38109469)

[7.1.2 Alerta registado na tabela de alertas 13](#_Toc38109470)

[7.1.3 Informação recebida no telemóvel 14](#_Toc38109471)

[7.1.4 Informação Adicional (se relevante) 14](#_Toc38109472)

[7.2 Situação 2 14](#_Toc38109473)

[7.2.1 Dados enviados pelo sensor 14](#_Toc38109474)

[7.2.2 Alerta registado na tabela de alertas 14](#_Toc38109475)

[7.2.3 Informação recebida no telemóvel 14](#_Toc38109476)

[7.2.4 Informação Adicional (se relevante) 14](#_Toc38109477)

[7.3 Situação 3 14](#_Toc38109478)

[7.3.1 Dados enviados pelo sensor 14](#_Toc38109479)

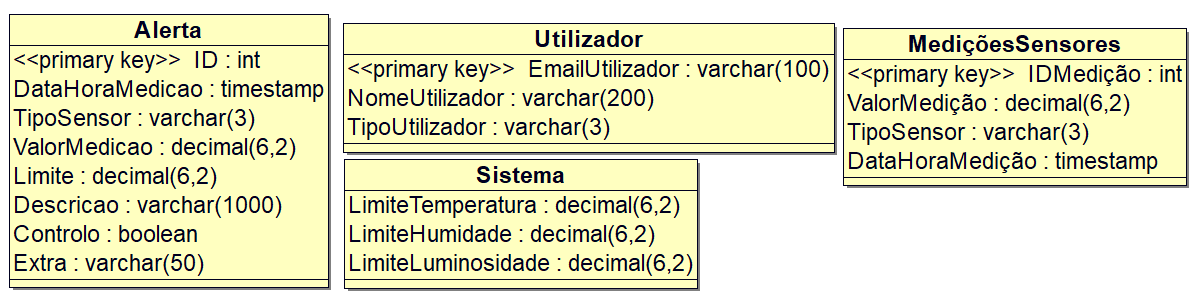
[7.3.2 Alerta registado na tabela de alertas 14](#_Toc38109480)

[7.3.3 Informação recebida no telemóvel 14](#_Toc38109481)

[7.3.4 Informação Adicional (se relevante) 14](#_Toc38109482)

Detecção de Intrusão e Incêndio em Museus

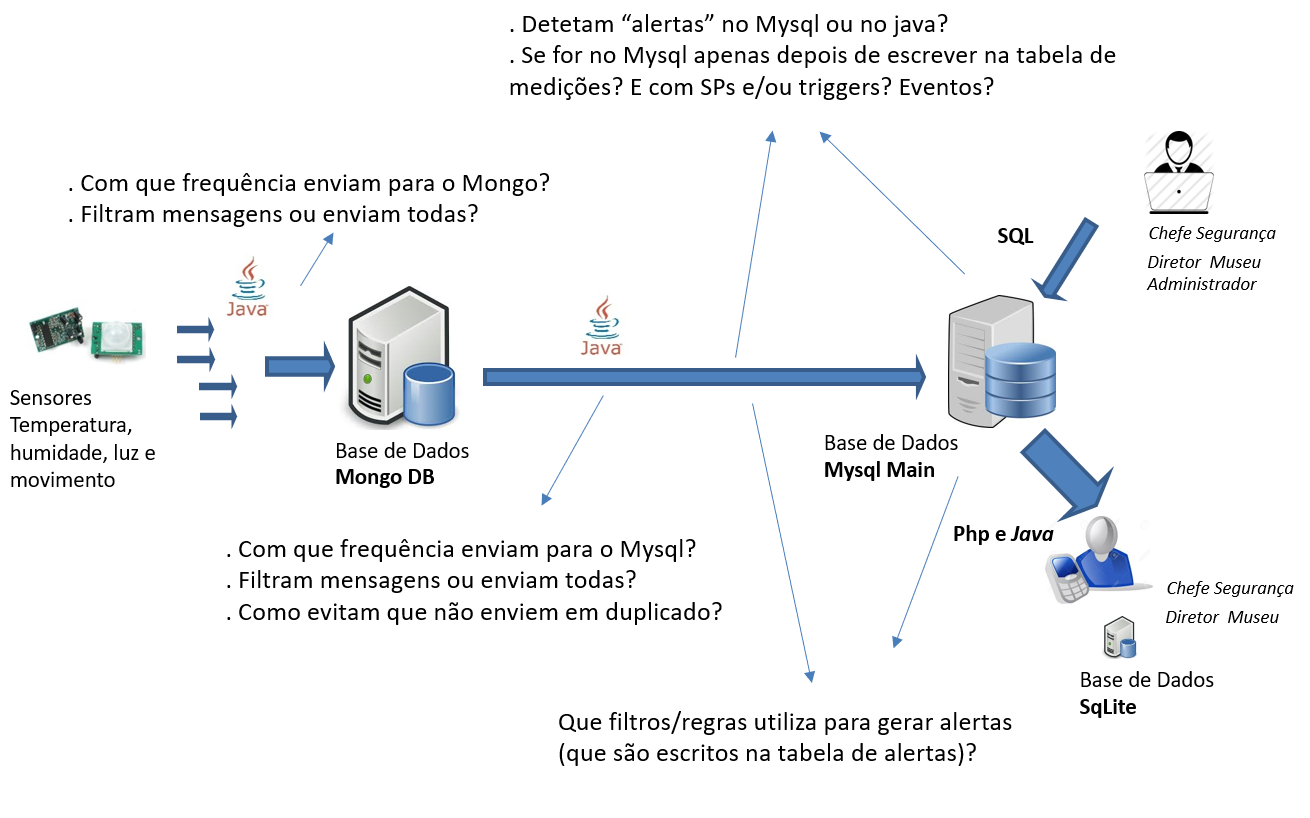
**Base de Dados MySQL**



**Sensor *Exemplo Mensagens***

{"tmp":"19.30","hum":"95.00","dat":"19/4/2020","tim":"9:50:51","cell":"228""mov":"0",”sens":"eth"} (\*)

**Esquema de Importação e Migração**



(\*) Enquanto o sensor não puder ser reparado, a mensagem é:

{"tmp":"19.30","hum":"95.00","dat":"19/4/2020","tim":"9:50:51","cell":"228""mov":"0""mov":"1""sens":"eth"} Para efeitos de trabalho removam **"mov":"0"** deve ser substituído por uma vírgula.

# Base de Dados Mongo

Nome bd: sensores

Nome Coleção1(\*): temperatura

Nome Coleção2(\*): humidade

Nome Coleção3(\*): luminosidade

Nome Coleção4(\*): movimento

Nome Coleção5(\*): erros

Exemplo documento coleção1:

{

"\_id": {

"$oid": "5e9f332beb64cb1834ae37be"

},

"tmp": "24.20",

"dat": "21/4/2020",

"tim": "17:53:49",

}

Exemplo documento coleção2:

{

"\_id": {

"$oid": "5e9f332beb64cb1834ae37be"

},

"hum": "95.00",

"dat": "21/4/2020",

"tim": "17:53:49",

}

Exemplo documento coleção3:

{

"\_id": {

"$oid": "5e9f332beb64cb1834ae37be"

},

"cell": "664",

"dat": "21/4/2020",

"tim": "17:53:49",

}

Exemplo documento coleção4:

{

"\_id": {

"$oid": "5e9f332beb64cb1834ae37be"

},

"mov": "1",

"dat": "21/4/2020",

"tim": "17:53:49",

}

Exemplo documento coleção5:

{

"\_id": {

"$oid": "5e9f332beb64cb1834ae37be"

},

"tmp": "?????",

"hum": "95.00",

"dat": "30/2/2020",

"tim": "17:53:49",

"cell": "664",

"mov": "1",

"sens": "eth"

}

Número de réplicas: 3

Optámos por uma arquitectura com três réplicas constituídas por um servidor primário e dois secundários. Apenas o servidor primário recebe os dados. Quando os dados são gravados no servidor primário este replica para os secundários. É possível ler de qualquer um dos três servidores. Se o servidor primário estiver indisponível um dos secundários assume o papel de primário.

(\*) O número de colecções é definido pelo grupo, pode optar só por uma ou por um qualquer número.

# Transporte Broker para Mongo (java)

## Periodicidade de Leitura de Sensores e Escrita no Mongo

<Explicar com que periodicidade o Java recebe informação dos sensores e envia para Mongo>

A informação gerada pêlos sensores é lida e gravada no Mongo com a periodicidade de 1 segundo.

## Utilização de Filtros que impeçam informação de ir para o Mongo

<Indicar se pretendem que o java implemente filtros que evitem que algum tipo de dados/situação vá para a base de dados MongoDB>

A informação recebida do broker é filtrada da seguinte maneira:

- Caso a mensagem não contenha erros os dados da temperatura serão gravados na colecção 1, os dados da humidade na colecção 2, os dados da luminosidade na colecção 3 e os dados do movimento na colecção 4.

- Caso exista algum problema com os dados a mensagem deve ser gravada na colecção 5 (erros). Para que seja possível analisar a posteriori problemas que estejam a ocorrer com os sensores. Mesmo com erros na mensagem, deve aproveitar-se toda a informação que seja coerente. Por exemplo se tivermos uma leitura com erro na temperatura devemos usar os valores dos outros sensores.

- Caso o problema seja na data ou hora não se gravam os dados dos sensores apenas na colecção 5 (erros).

Exemplos:

1 - Mensagem recebida: {"tmp":"24.20","hum":"75.5","dat":"21/4/2020",”tim": "17:53:49","cell":"664","mov":"1"}

Mensagem OK. Gravam-se os valores dos sensores nas respectivas colecções.

2 - Mensagem recebida: {"tmp":"24.20","hum":"zzzz","dat":"21/4/2020",”tim": "17:53:49","cell":"664","mov":"1"}

Mensagem com erro no valor da humidade. Grava-se a mensagem na colecção 5 (erros) e os valores dos sensores temperatura, luminosidade e movimento nas respectivas colecções.

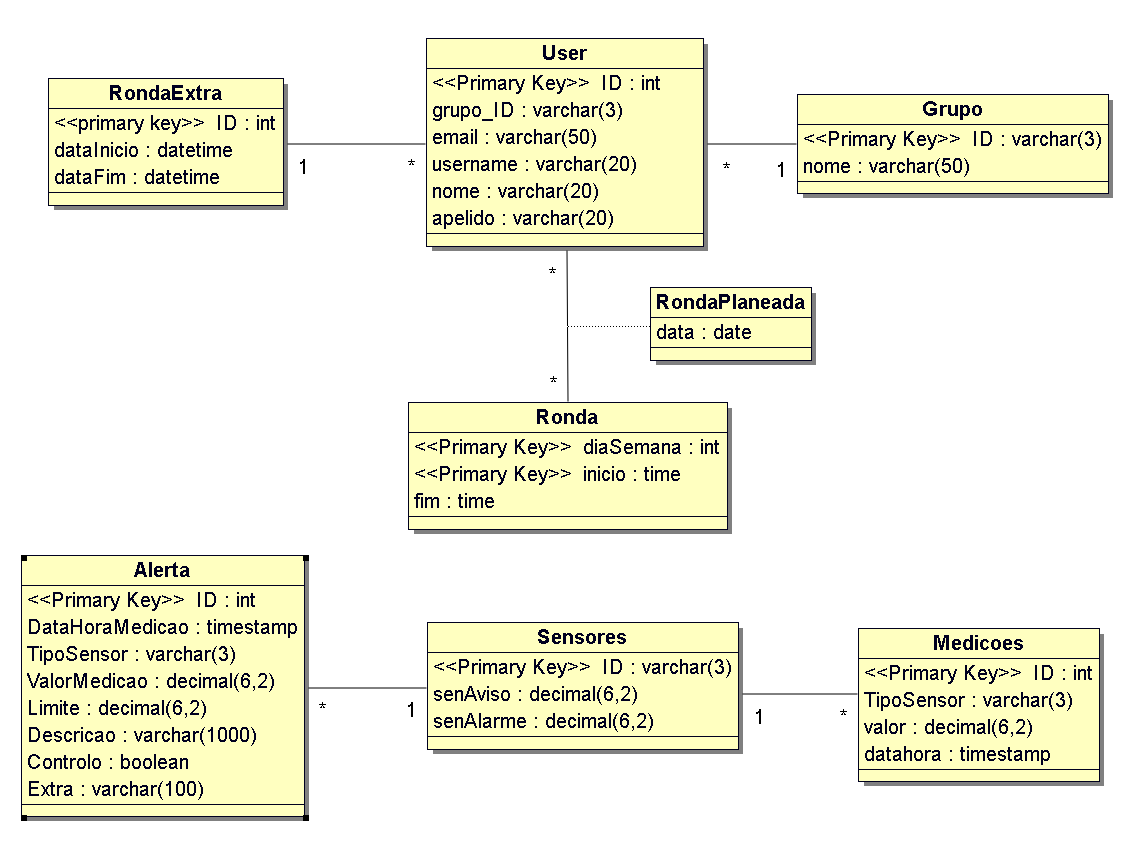
3 - Mensagem recebida: {"tmp":"24.20","hum":"75.5","dat":"00/00/0000",”tim": "00:00:00","cell":"664","mov":"1"}

Mensagem com erro na data ou hora. Grava-se a mensagem apenas na colecção 5 (erros).

# Estrutura da Base de Dados MySQL

## Diagrama relacional

<Diagrama relacional da base de dados MySQL. Podem efectuar alterações à base de dados apresentada no início do documento, no entanto, apenas os 8 campos da tabela Alerta indicados são reconhecidos pela aplicação Android Disponibilizada (os campos Controlo e Extra foram colocados apenas para antecipar eventuais necessidades dos grupos, não são processados pela aplicação Android.>



O campo **TipoSensor** da tabela **Alerta** é a chave estrangeira que faz a associação à tabela **Sensores**. Na tabela **Medicoes** a chave estrangeira é o campo **TipoSensor**. A tabela **Sensores** tem toda a informação para cada sensor (temperatura, humidade, movimento e luminosidade) valor de alerta, valor de aviso, etc. Optamos por esta implementação pois permitirá um número ilimitado de sensores.

## Privilégios de Utilizadores

<Quadro com tipos de utilizadores e indicação de privilégios sobre tabelas e SPs. Apenas utilizadores relevantes para esta fase>

O utilizador especial **system** com uma palavra-chave que apenas a equipa de desenvolvimento conhece (atenção! não é utilizador da aplicação) é o utilizador usado nas migrações.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Tabela | Tipo de utilizador | | | |
| Administrador | Director de Museu | Chefe de Segurança | system |
| Alerta | - | L | L | E |
| Sensores | E/L |  |  | L |
| Medicoes | - | L | L | E/L |
| User | E/L | - | - | - |
| Grupo |  |  |  | - |
| Ronda | - | - | E/L | L |
| RondaPlaneada | - | - | E/L | L |
| RondaExtra | - | - | L | L |

# Transporte Mongo para MySQL (java)

## Periodicidade de Leitura de Mongo e Escrita no MySQL

<Explicar com que periodicidade o Java recebe transporta informação para o MySQL>

O informação é lida do MongoDB e gravada no MySQL a cada 5 segundos.

## Utilização de Filtros que impeçam informação de ir para o MySQL

<Indicar se pretendem que o java implemente filtros que evitem que algum tipo de dados/situação vá para a base de dados MySQL (tabela Medições)>

Os valores são filtrados segundo a técnica:

O valor gravado no MySQL é a media dos valores com data posterior à data da última medição no MySQL. Para a temperatura, humidade e luminosidade valores que difiram do valor anterior em mais de 20% (este valor poderá ser alterado para melhor se adaptar ao funcionamento real dos sensores) são descartados e não vão ser considerados para a média. Assim evitamos valores anómalos como picos gerados pelos sensores. Para a data de medição é considerada a data e hora do valor mais recente no MongoDB.

Exemplo1: Como os dados dos sensores são criados no MongoDB a cada segundo e lidos a cada 5 segundos, a cada leitura existem 5 documentos no MongoDB. Para a tabela de medições o valor a considerar é a média dos 5 documentos com a data do documento mais recente.

Exemplo2: Supondo que temos 5 valores para a humidade {70,75,0,76,75} o valor 0 é descartado pois o valor não pode ser inferior 60 (75-15) a nem superior a 90 (75+15). Para a média contam apenas as leituras {70,75,76,75}.

Exemplo3: Supondo que temos 5 valores para a temperatura {18,20,99,22,24} o valor 99 é descartado pois o valor não pode ser inferior 16 (20-4) a nem superior a 24 (20+4).

## Mecanismo para assegurar que a migração é incremental

<Indicar de que forma garantem que não vai informação duplicada para a base de dados MySQL (tabela Medições)>

A duplicação de dados será evitada pela data e hora do documento. Para cada sensor, apenas são considerados os documentos com data e hora posterior à última medição no MySQL.

# Geração de Alertas

## Local/Momento de Detecção

Assinalar com X as situações pretendidas (podem ser várias)

No transporte Java \_

No MySQL antes de inserir na tabela de medições \_

De que forma: SP \_ Trigger: \_\_

No MySQL depois de inserir na tabela de medições **X**

De que forma: SP \_ Trigger: **X**

Recorrer a eventos:

## Regras para detecção de alertas

<Explicar de forma muito clara que regras devem ser implementadas, que tipos de alerta elas despoletam, de que forma os alertas são guardados na tabela alertas. Obrigatório dar exemplos, quer de dados que dão origem aos alertas, quer dos registos que são escritos na tabela alertas. De preferência recorrer a diagramas/ilustrações. Devem indicar explicitamente o nome dos sps que pretendem e/ou o nome das funções java.>

Para possibilitar ter um aviso antecipado quando um sensor se aproxima de um valor de alarme podemos especificar (no campo **senAviso** da tabela **Sensores**) para os sensores de temperatura e humidade, um valor de AVISO. Por exemplo para a humidade podemos ter como valor de AVISO 45%. Assim que o sensor passar dos 45% recebemos um aviso antecipado do aumento da humidade antes deste se tornar problemático. A diferença entre os registos de aviso e de alarme é o campo **Descricao** na tabela **Alerta** (Ver exemplo na tabela 1).

Regras para a Temperatura/Humidade:

- Caso a temperatura/humidade seja superior ao valor de aviso (**senAviso**) existe um Trigger after insert (**Medicoes\_insert**) na tabela **Medições** que cria um registo de aviso na tabela **Alerta** (Ver tabela 1).

- Caso a temperatura/humidade seja superior ao valor de alarme (**senAlarme**) para o respectivo sensor o Trigger **Medicoes\_insert** cria um registo de alarme na tabela **Alerta** (Ver tabela 1).

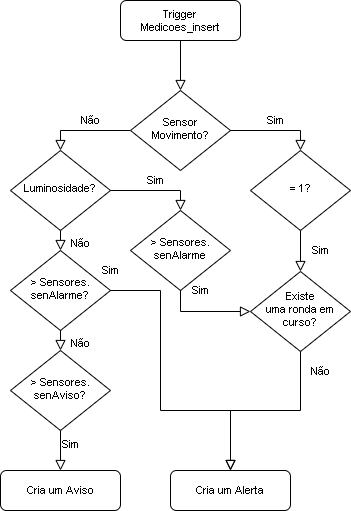
Regras para a Luminosidade:

Caso o valor da luminosidade seja superior ao valor de alarme (**senAlarme**) para o respectivo sensor, o Trigger **Medicoes\_insert** verifica se existe alguma ronda planeada ou extra prevista para a hora em que foi registada a leitura e caso negativo cria um registo de alarme na tabela **Alerta** (Ver tabela 1).

Regras para o Movimento:

Caso o valor do sensor de movimento seja igual ao valor de alarme (**senAlarme**), o Trigger **Medicoes\_insert** verifica se existe alguma ronda planeada ou extra prevista para a hora em que foi registada a leitura e caso negativo cria um registo de alarme na tabela **Alerta** (Ver tabela 1).

**Figura 1**



**Tabela 1**

|  |  |
| --- | --- |
| Campo na tabela Alerta | Valor do campo |
| DataHoraMedicao | Medicoes. datahora |
| TipoSensor | Sensores.senTipo |
| ValorMedicao | Medicoes. valor |
| Limite | Sensores. senAlarme ou Sensores.senAviso |
| Descricao | Ex.  “Alarme de Temperatura”  “Aviso de Humidade”. |
| Controlo | - |
| Extra | - |

|  |
| --- |
| Avaliação Global da Qualidade das Especificações recebidas  Avaliação (A,B,C,D,E) : \_\_\_\_\_\_\_\_\_\_\_\_\_  Utilize a seguinte escala:  A: - 1 – 5 valores B: 6 – 9 valores C: 10 – 13 Valores D: 14 – 17 valores E: 18 – 20 valores  **Análise crítica (clareza, completude, rigor):** |

# Implementação

## Divergências face ao recebido/especificado

<Indicar as divergências relevantes (ignorar pequenos detalhes de implementação) face ao especificado pelo próprio grupo e face ao especificado pelo outro grupo, nomeadamente as que consideram que permitiu chegar a uma solução melhor.>

### Ideias que aproveitaram do outro grupo

<Descrição clara, utilizar bullets.>

### Ideias que abandonaram do próprio grupo

<Descrição clara, utilizar bullets.>

### Novas ideias introduzidas

- A periodicidade de leitura do broker…

- Na migração do mongo para o MySQL descartamos os valores com data posterior à actual.

<Descrição clara, utilizar bullets.>

### Código Mongo Implementado (dentro do java)

<Listar todo o código Mongo utilizado dentro do java, quer para importar, quer para exportar. O código tem de ser comentado para que se torne legível para quem sabe uns rudimentos de MongoDB. Não é para colocar código java (excepto se for necessário para perceber o código mongo.>

# Demonstração

<Ilustrar o funcionamento da aplicação preferencialmente com capturas de imagens. Os exemplos têm de ilustrar todo o tipo de alertas, com a indicação dos respectivos tempos e da forma como visualmente a informação chega ao telemóvel.>

## Situação 1

Tipo de Alerta: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Dados enviados pelo sensor

### Alerta registado na tabela de alertas

<indicar também o tempo entre o envio do primeiro registo do sensor e o registo na tabela de alertas>

### Informação recebida no telemóvel

### Informação Adicional (se relevante)

## Situação 2

Tipo de Alerta: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Dados enviados pelo sensor

### Alerta registado na tabela de alertas

<indicar também o tempo entre o envio do primeiro registo do sensor e o registo na tabela de alertas>

### Informação recebida no telemóvel

### Informação Adicional (se relevante)

## Situação 3

Tipo de Alerta: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Dados enviados pelo sensor

### Alerta registado na tabela de alertas

<indicar também o tempo entre o envio do primeiro registo do sensor e o registo na tabela de alertas>

### Informação recebida no telemóvel

### Informação Adicional (se relevante)

…