

Sistemas de Informação Distribuídos













Licenciaturas em Engenharia Informática e Informática e Gestão de Empresas
2018-2019, Segundo Semestre

Monitorização de Culturas em Laboratório

Mongo DB e Android

Identificação do grupo autor
da especificação (Etapa A):

Identificação do grupo autor da
implementação (Etapas B):

| Número | Nome | Foto | Número | Nome | Foto |
|--------|-------------------|---|--------|----------------------|---|
| 60921 | Wang Yang |  | 77981 | André Silva |  |
| 72783 | André Carvalho |  | 73538 | Gonçalo Fernandes |  |
| 73353 | Bruno Colaço |  | 77812 | João Aparício |  |
| 74455 | Dinis Ferreira |  | 72809 | João Neto |  |
| 77667 | Maria Dinis |  | 77561 | João Saramago |  |
| 78014 | Mariana Silva |  | 77778 | Rita Costa |  |

Instruções

Estas instruções são de cumprimento obrigatório. Relatórios que não cumpram as indicações serão penalizados na nota final.

- Podem (e em várias situações será necessário) ser adicionadas novas páginas ao relatório, mas não podem ser removidas páginas. Se uma secção não for relevante, fica em branco, não pode ser removida;
- Todas as secções têm que iniciar-se no topo de página (colocar uma quebra de página antes);
- A paginação tem de ser sequencial e não ter falha;
- O índice tem de estar atualizado;
- Na folha de rosto (anterior) têm de constar toda a informação solicitada, nomeadamente todas as fotografias de todos os elementos dos dois grupos. É obrigatório que caiba tudo numa única página;
- A formatação das “zonas” (umas sombreadas outras não sombreadas) não pode ser alterada;
- O grupo que primeiro edita o documento (Etapa A) apenas escreve até à secção 1.7, e o outro grupo apenas em todas as outras secções.

Índice

| | | |
|-------|--|----|
| 1 | Mongo DB..... | 7 |
| 1.1 | Descrição Geral do Procedimento..... | 7 |
| 1.2 | Estrutura da Base de Dados Mongo..... | 12 |
| 1.3 | Periodicidade de Leitura de Sensores e Escrita no Mongo..... | 13 |
| 1.4 | Estrutura da Base de Dados Mysql..... | 14 |
| 1.5 | Periodicidade de Leitura de Mongo e Escrita no MySql | 15 |
| 1.6 | Triggers, SP ou eventos no MySql (caso relevante) | 16 |
| 1.7 | Utilizadores relevantes no Mysql e respectivos privilégios | 18 |
| | Avaliação Global da Qualidade das Especificações | 19 |
| 1.8 | Implementação | 20 |
| 1.8.1 | Divergências face ao recebido/especificado..... | 20 |
| 1.8.2 | Código Mongo Implementado (dentro do java) | 22 |
| 1.8.3 | Código SQL Implementado..... | 24 |
| 1.8.4 | Tempo Médio | 27 |
| 1.8.5 | Alertas | 28 |
| 2 | Android e Php..... | 31 |
| 2.1 | Esquema da BD Lite Geral | 31 |
| 2.2 | Layout Implementado no Android | 32 |

Monitorização de Culturas em Laboratório

Um laboratório de investigação de um departamento de biologia necessita de um sistema para monitorizar a evolução de culturas. Mais concretamente, pretende acompanhar a temperatura e luz a que as culturas estão sujeitas, bem como detectar/antecipar potenciais problemas.

Numa estufa estão colocados dois sensores que medem a temperatura e quantidade de luz ambiente (que afeta todas as culturas existentes na estufa).

Periodicamente os investigadores dirigem-se à estufa para efetuarem manualmente várias medições de variáveis (humidade, ph, etc) e registá-las num computador que está localizado na estufa. Cada cultura tem um único investigador responsável e apenas ele pode criar, atualizar e consultar os dados de medições das suas culturas. Esta *proteção de dados* é um aspeto importante do sistema. Nem todas as variáveis necessitam serem lidas e registadas. Para cada cultura o investigador decide quais delas devem ser lidas, e regista no sistema qual o intervalo de valores que considera “normal” para o par variável/cultura.

Por exemplo, para as culturas hidropónicas de pimento e tomate, fazem-se medições do nível de concentração de mercúrio e chumbo. Mas numa cultura de bactérias onde se adicionaram antibióticos o que faz sentido medir é o índice de concentração das bactérias, não faz sentido medir o nível de concentração de mercúrio e chumbo.

Alertas

Existem dois tipos de alertas:

a) alertas resultantes das medições das variáveis. O investigador, quando insere manualmente um valor de uma medição, caso o valor ultrapasse os limites será alertado com um aviso (no próprio computador) e com uma mensagem para o telemóvel (por vezes o investigador pede a um colega para efectuar a medição, sendo por isso aconselhável que o alerta não apareça somente no monitor do computador).

b) Alertas resultantes dos sensores de temperatura e luminosidade. O sistema sabe, para toda a estufa, o intervalo de valores de luminosidade e temperatura adequado (igual para todas as culturas). Se o sensor detectar que os valores vão ser ultrapassados deve notificar por telemóvel o investigador.

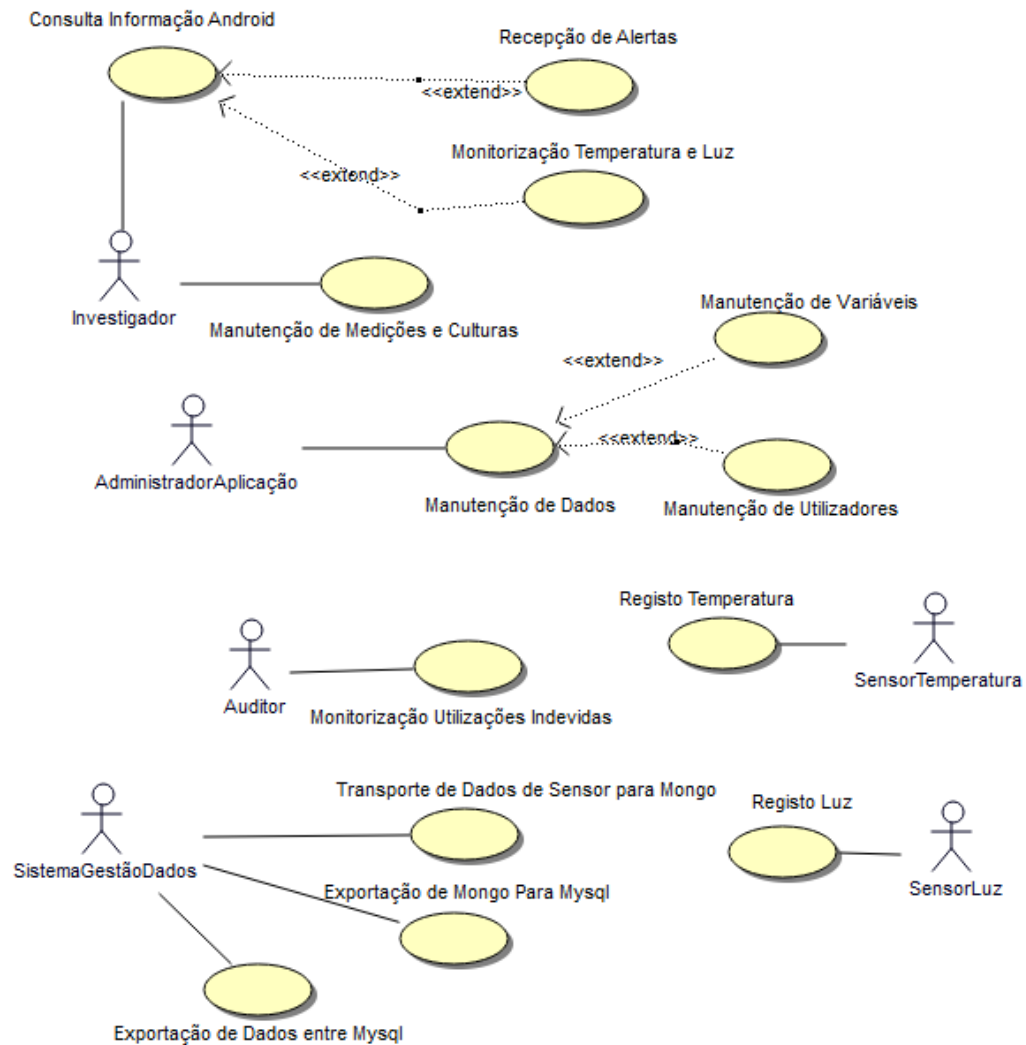
Cada investigador deverá ter a possibilidade de, através de um telemóvel, monitorizar a evolução da temperatura e luminosidade (não apenas a última leitura, mas a evolução na última hora ou horas) e receber os dois tipos de alertas.

Registo de Acessos

É necessário guardar na base de dados (mysql) o registo de todas as operações de escrita sobre todas as tabelas (quais dados foram alterados/inseridos/apagados, quando e por quem) e o registo de operações de consulta apenas sobre a tabela Medições. Esse registo de alterações (*log*) é *exportado* incrementalmente (apenas informação nova) e periodicamente para uma base de dados autónoma (também mysql). Através dessa base de dados (apenas de consulta) um

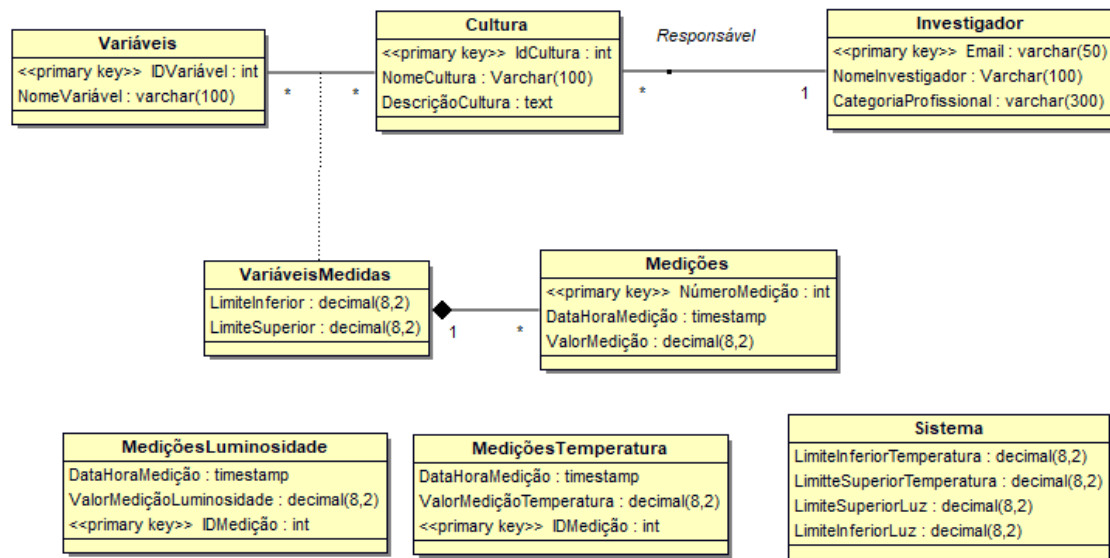
auditor pode analisar se ocorreram utilizações abusivas dos dados (por exemplo, quem é que alterou limites de temperatura de uma cultura, etc.).

Diagrama de Use Case Global



No presente relatório apenas são contemplados os use case “Registo Temperatura”, “Registo Luz”, “Consulta Informação Android”, “Transporte de Dados de Sensor Para Mongo”, “Exportação de Mongo para MySql” e “Exportação de Dados entre Mysql”.

Diagrama de Classes de Suporte à Base de Dados

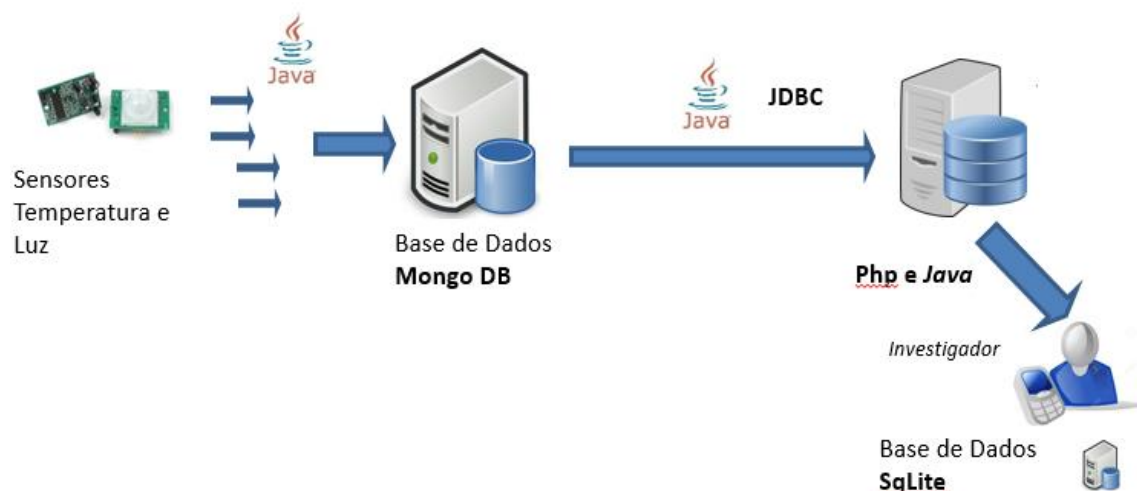


Sensor

Exemplo Mensagens

```
{"tmp":"22.40","hum":"61.30","dat":"9/4/2019","tim":"14:59:32","cell":"3138","sens":"wifi"}
```

Esquema de Importação e Migração



1 Mongo DB

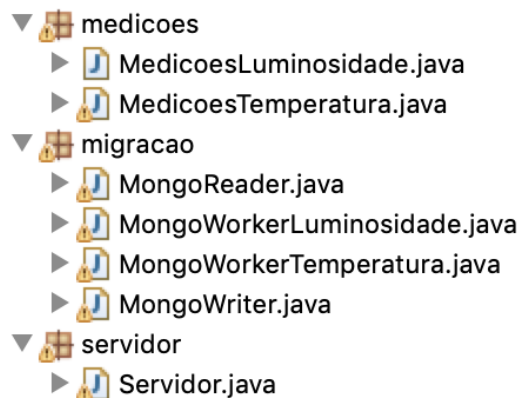
1.1 Descrição Geral do Procedimento

Nesta secção serão especificados os elementos relativos ao processo de migração dos dados entre a base de dados MongoDB e o MySQL.

Para a implementação desta fase da aplicação o grupo deve utilizar as bibliotecas citadas abaixo:

- mongodb-driver-3.4.3.jar
- org.eclipse.paho.client.mqttv3-1.2.0.jar
- mysql-connector-java-8.0.15.jar

O projeto deve encontrar-se estruturado como representado abaixo, as restantes informações serão explicitadas de seguida:



Primeiramente deve ser tratada a questão relativa à receção de dados do sensor e a forma como estes devem ser guardados na base de dados MongoDB.

As classes *MedicoesLuminosidade* e *MedicoesTemperatura* são classes bastantes simples e devem ter os construtores citados abaixo, bem como funções *get* e *set* que podem ser úteis posteriormente:

```
public MedicoesLuminosidade(String id, double valor, Date data)
    public MedicoesLuminosidade(double valor, Date data)
    public MedicoesTemperatura(String id, double valor, Date data)
    public MedicoesTemperatura(double valor, Date data)
```

Deve optar-se por criar a classe *Servidor* (com main) que implementa a interface *Mqttcallback*. De seguida, deve ser criado o método *messageArrived* conforme explicitado abaixo:

```
public void messageArrived(String topic, MqttMessage message)
```

Este método tem como objetivo tratar as mensagens recebidas do sensor (note-se que se pode utilizar o *paho* de forma a simular o comportamento do sensor), que envia mensagens com o seguinte formato:

```
{"tmp":"22.40","hum":"61.30","dat":"9/4/2019","tim":"14:59:32","cell":"3138","sens":"wifi"}
```

Face aos dados recebidos devemos isolar os componentes essenciais, que no caso são os valores relativos à temperatura e luminosidade, bem como, a data da medição. Esta ação pode ser realizada com recurso à função Java *split* ou qualquer outra equivalente. De seguida, devem ser criados os objetos *MedicoesTemperatura* e *MedicoesLuminosidade* com o valor da medição e data (rever construtores citados acima).

As anomalias serão filtradas antes de os respetivos dados serem inseridos na base de dados MongoDB, ou seja, na classe *Servidor* com recurso aos métodos *filtrarLuminosidade* e *filtrarTemperatura*, tanto nas medições de luminosidade como nas medições de temperatura, embora tenham critérios distintos que explicaremos de seguida.

O sensor de temperatura envia leituras em graus centígrados. Deve ser criada uma *ArrayBlockingQueue* com o tamanho 10, como exemplificado em baixo. Assim, consegue-se garantir que a leitura de temperatura recebida do sensor é comparada não só com os valores anteriores, como também com os valores seguintes.

```
ArrayBlockingQueue<MedicoesTemperatura> medicoesTemp = new ArrayBlockingQueue<MedicoesTemperatura>(10);
```

No método *filtrarTemperatura* devem ser tratadas as anomalias. Cria-se um vetor de *MedicoesTemperatura* onde são colocados os valores da *BlockingQueue*, conforme especificado abaixo. Sempre que uma nova medição é adicionada ao vetor anterior é criado um vetor ordenado.

```
MedicoesTemperatura[] medT = new MedicoesTemperatura[10];
medicoesTemp.toArray(medT);
MedicoesTemperatura[] medTOrdenado = new MedicoesTemperatura[10];
```

A ordenação pode ser realizada com auxílio da função *sort* ou qualquer outra equivalente, fica ao critério do grupo. Este vetor *medTOrdenado* terá como função o cálculo da mediana das 10 leituras e, assim, decidir se a leitura de índice 5 do vetor *medT* é, ou não, uma anomalia. Deve considerar-se um desvio padrão de 5. Ver exemplo abaixo:

| | | | | | | | | | | |
|------|----|----|----|----|----|----|----|----|----|----|
| medT | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | 20 | 18 | 19 | 20 | 20 | 40 | 20 | 17 | 20 | 30 |

| | | | | | | | | | | |
|--------------|----|----|----|----|----|----|----|----|----|----|
| medTOrdenado | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | 17 | 18 | 19 | 20 | 20 | 20 | 20 | 20 | 30 | 40 |

Média = 20
40 pertence ao intervalo de 15 a 25? Não, **DESCARTAR!**

| | | | | | | | | | | |
|------|----|----|----|----|----|----|----|----|----|----|
| medT | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | 20 | 18 | 19 | 20 | 20 | 23 | 20 | 17 | 20 | 30 |

| | | | | | | | | | | |
|--------------|----|----|----|----|----|----|----|----|----|----|
| medTOrdenado | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| | 17 | 18 | 19 | 20 | 20 | 20 | 20 | 23 | 30 | 40 |

Média = 20
23 pertence ao intervalo de 15 a 25? Sim, **ACEITAR!**

O sensor de luminosidade envia leituras na unidade de medida *lux*. Neste caso, a detecção de anomalias é mais simples. Ao contrário da temperatura, em que alterações bruscas são anomalias do sensor, neste caso, estas podem ocorrer. Não há muita concordância entre as fontes no que toca ao intervalo de valores possível de luminosidade, assim, decidimos guiar-nos pela seguinte tabela:

| Natural Light Condition | Typical Lux |
|-------------------------|-------------------|
| Direct Sunlight | 32,000 to 100,000 |
| Ambient Daylight | 10,000 to 25,000 |
| Overcast Daylight | 1000 |
| Sunset & Sunrise | 400 |
| Moonlight (Full moon) | 1 |
| Night (No moon) | < 0.01 |

Desta forma, todos os valores inferiores ou iguais a 0 devem ser descartados, bem como todos os que ultrapassem 100.000 visto que se tratam de claros erros de leitura do sensor. A verificação é feita desta forma uma vez que, por exemplo, tem que ser aceite que a luz de uma estufa apague e acenda num curto período de tempo, o que se traduz numa alteração brusca que não pode ser considerada uma anomalia. Só os valores pertencentes ao intervalo possível devem ser escritos na base de dados MongoDB.

No método *filtrarLuminosidade* devem ser tratadas as anomalias. Cria-se um vetor de *MedicoesLuminosidade* onde são colocados os valores da *BlockingQueue*, conforme especificado abaixo (semelhante à temperatura):

```
MedicoesLuminosidade[] medL = new MedicoesLuminosidade[10];
medicoesLum.toArray(medL);
```

Após filtrar todas as anomalias será explicada a forma como os dados serão escritos na base de dados MongoDB. Para este fim, deve ser criada a classe *MongoWriter*. No construtor da respetiva classe deve ser feita a ligação ao Mongo obtendo-se a base de dados pretendida, conforme exemplificado abaixo:

```
public MongoWriter() {
    mongoClient = new MongoClient("localhost");
    db = mongoClient.getDB("sensores");
}
```

De seguida, deve ser criada o método *write* cuja função é escrever as medições na base de dados MongoDB:

```
public void write (MedicoesLuminosidade l, MedicoesTemperatura t)
```

As medições devem ser adicionadas às respetivas coleções, *medicoesluminosidade* e *medicoestemperatura*. A inserção deve ser realizada com recurso ao comando *put*. Pode ser visto abaixo um exemplo de como se pode obter a coleção pretendida:

```

DBCollection table = db.getCollection("medicoesluminosidade");
DBCollection table = db.getCollection("medicoestemperatura");

```

De modo a garantir que a conexão à base de dados é realizada apenas uma vez, na classe *Servidor* deve ser dado um objeto *MongoWriter* como atributo.

Para leitura da base de dados MongoDB será criada a classe *MongoReader* (com main) que se irá ligar à mesma da seguinte forma:

```

mongoClient = new MongoClient("localhost");
db = mongoClient.getDB("sensores");

```

Serão, também, aqui iniciadas duas threads cuja função será ler os dados das coleções *medicoesluminosidade* e *medicoestemperatura*. Foram então criadas as classes *MongoWorkerTemperatura* e *MongoWorkerLuminosidade* que recebem os seguintes argumentos:

```

public MongoWorkerTemperatura(DB db, ArrayList<MedicoesTemperatura> medicoesTemperatura) {
    this.db = db;
    this.medicoesTemperatura = medicoesTemperatura;
}
public MongoWorkerLuminosidade(DB db, ArrayList<MedicoesLuminosidade> medicoesLuminosidade) {
    this.db = db;
    this.medicoesLuminosidade = medicoesLuminosidade;
}

```

No método *run* de cada uma das threads deve ser feita a conexão ao MySQL:

```

Connection conn = getMySQLDataSource().getConnection();

```

Pode ser criada a função *getMySQLDataSource* (que devolve um objeto do tipo *Mysqldatasource*) de forma a dar os parâmetros necessários para a respetiva ligação, do seguinte modo:

```

dataSource.setServerName("localhost");
dataSource.setPortNumber(3306);
dataSource.setDatabaseName("dba");
dataSource.setUser("sensor");
dataSource.setPassword("password");
return dataSource;

```

O último ID exportado é sempre guardado nas coleções *idTemperatura* e *idLuminosidade*, por isso, antes de ser realizada a leitura das medições (explicitado em seguida), o último ID é lido da base de dados MongoDB e guardado numa variável *idMedicao*.

Visto que nestas classes a base de dados já foi recebida como argumento, o próximo passo será aceder à coleção que se pretende ler (já explicado anteriormente). De seguida, com recurso a cursor deve ser lido cada documento de cada uma das coleções: *medicoestemperatura* e *medicoesluminosidade*. Os valores lidos devem ser guardados:

```

double valor = (Double) objeto.get("ValorMedicao");
Date data = (Date) objeto.get("Data");

```

Uma vez que a ligação ao MySQL já foi feita, realiza-se um simples comando:

```
PreparedStatement preparedStatement = conn.prepareStatement("INSERT INTO ...")+"VALUES (?, ?, ?)");
preparedStatement.setInt(1, idMedicao);
preparedStatement.setTimestamp(2, t);
preparedStatement.setDouble(3, valor);
preparedStatement.executeUpdate();
```

Após a inserção anterior da linha na tabela do MySQL, deve ser feito um SELECT a essa mesma tabela de forma a garantir que o *idMedicao* foi registado com sucesso. Caso se verifique, o documento é inserido na coleção *backuptemperatura* (ou *backupluminosidade*, conforme o caso) e removido da *medicoestemperatura* (ou *medicoesluminosidade*). Assim, garante-se que não existe perda de dados e que a migração é realizada de forma incremental.

Seguidamente, o *idMedicao* é incrementado e, aquando a inserção do último registo, este é guardado na coleção *idTemperatura* (ou *idLuminosidade*). Relembra-se que este processo se mantém para os dois *Workers*.

De forma sistemática, devem ser utilizados dois main, um na classe *Servidor* e outro na classe *MongoReader*. Devem ser utilizadas apenas duas threads, a *MongoWorkerTemperatura* e a *MongoWorkerLuminosidade*.

1.2 Estrutura da Base de Dados Mongo

Nesta secção serão exemplificadas e explicadas em mais detalhe as coleções que devem ser utilizadas (e que já foram mencionadas anteriormente) na base de dados MongoDB *sensores*.

Deve optar-se por utilizar duas coleções principais e quatro coleções auxiliares. As coleções principais são a *medicoestemperatura* e *medicoesluminosidade* e são responsáveis por armazenar os dados das leituras enviadas pelo sensor. A informação encontra-se exemplificada abaixo:

```
> db.medicoestemperatura.find().pretty()
{
  "_id" : ObjectId("5cb73e0e4e417727e2d4f0e4"),
  "ValorMedicao" : 11.93,
  "Data" : ISODate("2019-05-09T13:59:32Z")
}
```

```
> db.medicoesluminosidade.find().pretty()
{
  "_id" : ObjectId("5cb73e0e4e417727e2d4f0ea"),
  "ValorMedicao" : 57948.51217219059,
  "Data" : ISODate("2019-05-09T13:59:32Z")
}
```

De seguida, serão utilizadas as coleções auxiliares *backuptemperatura* e *backupluminosidade* que têm como função guardar toda a informação que é migrada da base de dados MongoDB para o MySQL. São movidos, após cada exportação, das duas coleções principais para as citadas acima. A informação pode ser vista abaixo, com a mesma estrutura da anterior:

```
> db.backuptemperatura.find().pretty()
{
  "_id" : ObjectId("5cb73e0e4e417727e2d4f0e4"),
  "ValorMedicao" : 11.93,
  "Data" : ISODate("2019-05-09T13:59:32Z")
}
```

```
> db.backupluminosidade.find().pretty()
{
  "_id" : ObjectId("5cb73e0e4e417727e2d4f0ea"),
  "ValorMedicao" : 57948.51217219059,
  "Data" : ISODate("2019-05-09T13:59:32Z")
}
```

Por fim, as últimas duas coleções auxiliares são a *idtemperatura* e *idluminosidade* e são utilizadas apenas para registar o último ID exportado de cada uma das duas coleções principais:

```
> db.idtemperatura.find().pretty()
{ "_id" : ObjectId("5cb88fa7b9a810860e2191a7"), "ID" : 1 }
{ "_id" : ObjectId("5cb8b5807cce2b3693272773"), "ID" : 255 }
```

```
> db.idluminosidade.find().pretty()
{ "_id" : ObjectId("5cb88f9eb9a810860e2191a6"), "ID" : 0 }
{ "_id" : ObjectId("5cb8b5817cce2b3693272774"), "ID" : 409 }
```

1.3 Periodicidade de Leitura de Sensores e Escrita no Mongo

O sensor envia uma leitura a cada 2 segundos com o seguinte formato:

```
{"tmp":"22.40","hum":"61.30","dat":"9/4/2019","tim":"14:59:32","cell":"3138","sens":"wifi"}
```

Assim, devem inserir-se os dados na base de dados MongoDB sempre que são recebidos, visto que o volume de dados em cada mensagem é bastante reduzido. Devem, então, ser logo adicionados à coleção correspondente.

Foram realizados testes para a leitura e a escrita na base de dados MongoDB, cada leitura é escrita em cerca de aproximadamente 2.5 milissegundos. Este tempo médio não é afetado pela quantidade de dados recebida pelo sensor, visto que quando é recebida a medição seguinte, a anterior já foi escrita no Mongo.

Como mencionado na secção anterior, deverá existir uma coleção de backup que pode ser utilizada em caso de falhas, e não pode ser apagada.

1.4 Estrutura da Base de Dados Mysql

Nesta secção serão apresentadas as tabelas relevantes para esta fase, bem como alguns exemplos de informação guardada nas mesmas.

As tabelas relevantes encontram-se representadas abaixo:

| | |
|---------------------------------------|-----------------------|
| medicoesluminosidade | alertas |
| IDMedicao INT(11) | IDAlerta INT(11) |
| DataHoraMedicao DATETIME | Alerta VARCHAR(200) |
| ValorMedicaoLuminosidade DECIMAL(8,2) | DataHora DATETIME |
| Indexes | IDCultura INT(11) |
| Triggers | HoraConsulta DATETIME |
| | Indexes |

| |
|--------------------------------------|
| medicoestemperatura |
| IDMedicao INT(11) |
| DataHoraMedicao DATETIME |
| ValorMedicaoTemperatura DECIMAL(8,2) |
| Indexes |
| Triggers |

As tabelas *medicoesluminosidade* e *medicoestemperatura* já tinham sido criadas anteriormente. Devem ter como chave primária um ID, e devem guardar, também a data e a hora da medição, bem como o respetivo valor. Ver exemplo de informação abaixo:

| IDMedicao | DataHoraMedicao | ValorMedicaoTemperatura |
|-----------|---------------------|-------------------------|
| 1 | 2019-05-09 14:59:32 | 11.93 |

| IDMedicao | DataHoraMedicao | ValorMedicaoLuminosidade |
|-----------|---------------------|--------------------------|
| 1 | 2019-05-09 14:59:32 | 2618.60 |

A tabela *alertas* tem como chave primária o ID do alerta, guarda a data e a hora em que o alerta foi gerado, bem como a cultura (se existir), o alerta (em texto) e, por fim, a hora de consulta (se tiver sido consultado). Existem dois tipos de alertas, os resultantes das medições do sensor e os resultantes da inserção manual de uma medição. Encontram-se ambos exemplificados abaixo:

| IDAlerta | Alerta | DataHora | IDCultura | HoraConsulta |
|----------|---|---------------------|-----------|---------------------|
| 1 | A cultura com ID 3 recebeu um valor da 'variável' superior ao limite. | 2019-05-09 14:59:32 | 3 | NULL |
| 2 | O limite superior de temperatura da estufa foi ultrapassado. | 2019-05-09 14:59:32 | NULL | 2019-05-09 15:02:32 |
| 3 | O limite superior de temperatura da estufa foi ultrapassado. | 2019-05-09 15:45:32 | NULL | NULL |

O primeiro é um alerta resultante de uma medição manual e o segundo, de uma leitura do sensor. O campo *HoraConsulta* é preenchido apenas caso o investigador veja o alerta. O campo *IDCultura* deve ser preenchido apenas no caso das medições inseridas manualmente.

Sempre que o valor de uma das medições ultrapassa os limites estipulados, quer num caso, quer no outro, é gerado um alerta. A inserção, ou não, dessa linha na tabela *alertas* (que mais tarde gera uma notificação no telemóvel do investigador) será explicada adiante.

1.5 Periodicidade de Leitura de Mongo e Escrita no MySql

Nesta secção será explicado de que forma e com que periodicidade o Java recebe informação do Mongo e exporta para o MySQL.

Se for seguida a especificação anterior o Java leva cerca de 420 milissegundos a aceder à coleção pretendida, valor obtido através de testes. O tempo de leitura do Mongo e escrita no MySQL (migração) é de aproximadamente 76 milissegundos.

Visto que o sensor envia dados de 2 em 2 segundos e que se deve garantir, acima de tudo, que o investigador pode ter acesso às informações o mais rapidamente possível, o processo acima deve ocorrer com uma periodicidade de 3 em 3 segundos (para ter uma margem de segurança face ao valor anterior).

A forma como a informação é recebida pelo Java encontra-se explicitada no tópico 1.1.

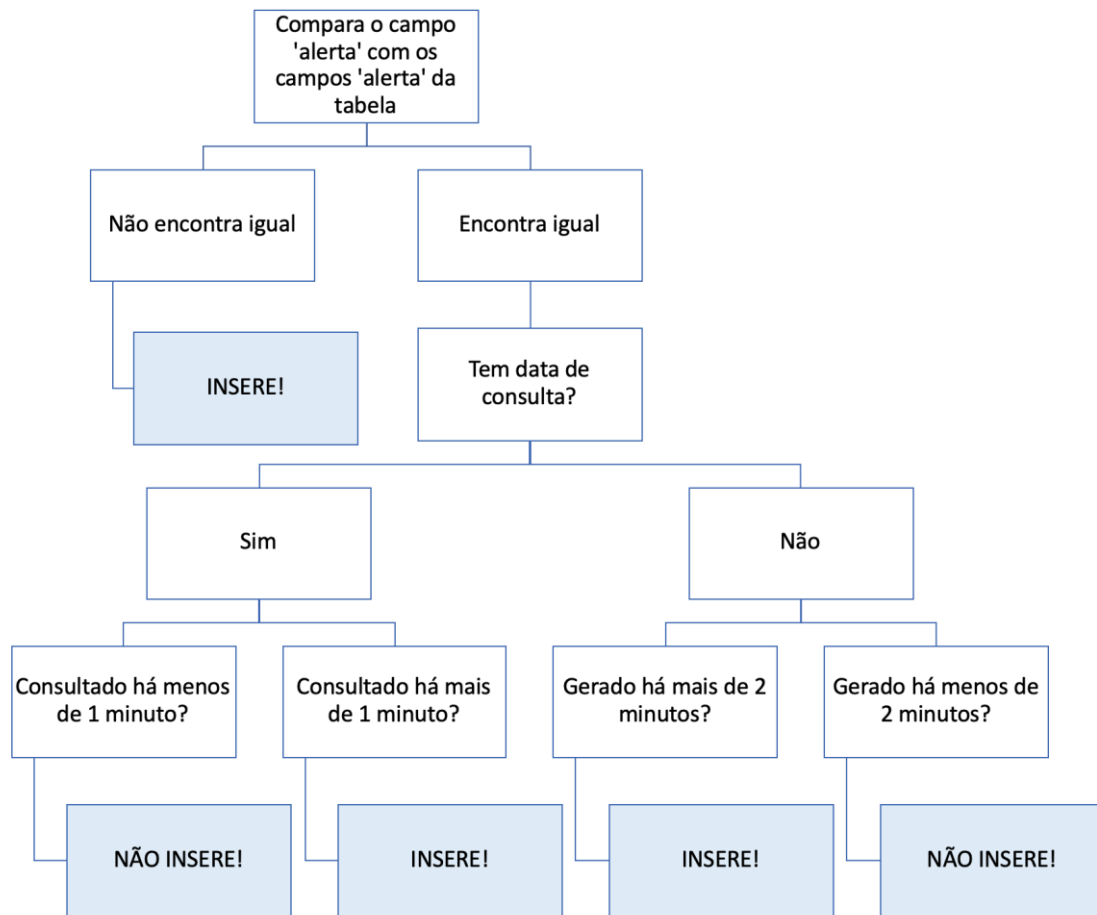
1.6 Triggers, SP ou eventos no MySql (caso relevante)

Encontram-se abaixo especificados os procedimentos e os triggers que devem ser implementados. Cada investigador deverá ter a possibilidade de monitorizar a evolução da temperatura e luminosidade na última hora, ou horas.

| Nome Procedimento | Parâmetros Entrada | Parâmetros Saída | Muito breve descrição |
|--------------------|--------------------|-------------------------------|--|
| selectTemperatura | Número de horas | ValorMedicao, DataHoraMedicao | Devolve a tabela <i>medicoestemperatura</i> , de forma a permitir a monitorização da mesma por parte do investigador. |
| selectLuminosidade | Número de horas | ValorMedicao, DataHoraMedicao | Devolve a tabela <i>medicoesluminosidade</i> , de forma a permitir a monitorização da mesma por parte do investigador. |
| updateAlerta | DataHoraConsulta | - | Atualiza a linha na tabela <i>alertas</i> com a hora em que o investigador viu o alerta. |

| Nome Trigger | Tabela | Tipo de Operação (I,U,D) | Evento (A, B) | Notas (apenas indicar aquilo que não seja óbvio) |
|--------------------|-----------------------|--------------------------|---------------|---|
| alertaTemperatura | medicoes temperatura | I | A | Se é inserida uma medição que ultrapasse os valores definidos para a estufa, é gerado um alerta, inserindo uma nova entrada na tabela <i>alertas</i> . |
| alertaLuminosidade | medicoes luminosidade | I | A | Se é inserida uma medição que ultrapasse os valores definidos para a estufa, é gerado um alerta, inserindo uma nova entrada na tabela <i>alertas</i> . |
| alertaMedicoes | medicoes | I | A | Se é inserida uma medição que ultrapasse os valores definidos para a cultura, é gerado um alerta, inserindo uma nova entrada na tabela <i>alertas</i> . |
| filtrarAlerta | alertas | I | B | Deve filtrar os alertas conforme explicitado no esquema abaixo. |

O trigger *filtrarAlerta* deve filtrar a entrada na tabela *alertas* da seguinte forma, explicitada abaixo:



1.7 Utilizadores relevantes no Mysql e respectivos privilégios

Nesta secção serão explicitados os utilizadores e respetivas permissões sobre as tabelas e stored procedures relevantes para esta fase.

| Tabela | Utilizadores | | |
|--------------------------|--------------|--------------|---------------|
| | Sensor | Investigador | Administrador |
| medicoesluminosidade | E, L | - | L |
| medicoestemperatura | E, L | - | L |
| alertas | - | L | L |
| Stored Procedures | | | |
| selectTemperatura | - | X | - |
| selectLuminosidade | - | X | - |
| updateAlerta | - | X | - |

Avaliação Global da Qualidade das Especificações recebidas

Avaliação (A,B,C,D,E) : D

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):

Consideramos que a especificação recebida é bastante completa, porém não abrange todos os casos de erro que podem decorrer das leituras dos sensores, como por exemplo no caso da luminosidade vir errada o erro nunca é detetado e todas as medições são consideradas corretas desde que tenham valores dentro do espectro de valores possíveis de luminosidade.

No caso da temperatura a estratégia adotada de recorrer a uma mediana para verificar erros seria adequada se o valor de variação para a mesma fosse parametrizável (é usado o valor 5 mas os investigadores podem não concordar com o mesmo),podendo dar-se o caso de os valores limite da temperatura não diferirem em 10 graus entre si.

Consideramos que a estratégia adotada poderia ser mais rigorosa se os valores das medições considerados errados não fossem descartados mas sim guardados numa tabela de erros ou marcados como errados com o intuito de verificar o mau funcionamento dos sensores.

Quanto à estrutura da base de dados mongo, pensamos que não é necessário ter duas coleções distintas para a temperatura e a luminosidade, uma vez que os campos são idênticos.

Como o mongo é utilizado apenas como um meio de transporte não consideramos necessário fazer backups dos dados migrados.

1.8 Implementação

1.8.1 Divergências face ao recebido/especificado

A principal diferença entre a especificação recebida e a nossa especificação é a forma como é utilizada a base de dados mongo.

No nosso caso existe apenas uma coleção onde são registadas as medições, podendo nelas constar os seguintes campos: timestamp, temperatura, luminosidade, alertaTemperatura, alertaLuminosidade, erroTemperatura, erroLuminosidade, exportado. De referir que os campos temperatura, luminosidade, alertaTemperatura, alertaLuminosidade, erroTemperatura, erroLuminosidade apenas são inseridos na base de dados mongo caso hajam esses mesmos valores na nova medição. A verificação de erros e alertas é feita no java.

Na especificação recebida são usadas coleções diferentes para as medições de temperatura e de luminosidade e a verificação dos respetivos alertas é feita através de triggers. Na especificação recebida são guardadas as medições já exportadas para o relacional o que consideramos ser desnecessário visto que o mongo serve apenas de transporte.

- i) Da nossa especificação foi utilizada na implementação o nosso mecanismo de deteção de erros e de alertas por considerarmos que é mais completo e parametrizável, uma vez que as percentagens consideradas para a deteção de erros e de alertas ficam ao cargo dos investigadores.

A estrutura da base de dados mongo também foi a que especificámos, sendo que acrescentámos os campos 'causaTemperatura' e 'causaLuminosidade', onde é colocada a descrição dos alertas. Consideramos que a nossa solução é mais adequada, uma vez que usa as funcionalidades de uma base de dados mongo, ao contrário da especificação recebida que tem várias coleções à semelhança do que é feito com as tabelas em relacional.

- ii) Tal como a especificação recebida, decidimos apagar as medições da base de dados mongo depois de ser feita a exportação para o relacional. Porém ao contrário do especificado não guardamos esses dados

numa tabela de backup, pois consideramos que o mongo serve apenas de transporte e não há qualquer vantagem em guardar esses dados.

Utilizámos também na nossa implementação a verificação da medição de luminosidade vir com um valor inferior a zero para detetar erros.

Seguimos a estratégia de ter dois mains uma vez que a mesma traz uma maior escalabilidade e versatilidade à aplicação.

- iii) Decidimos, alterar em relação ao que tínhamos especificado, a permanência dos dados na base de dados mongo: na implementação, quando a exportação é feita com sucesso para o relacional, os dados são apagados. Tomámos esta decisão, uma vez que o mongo apenas serve de meio de transporte e não há nenhum benefício em que os dados continuem lá.

Devido ao anteriormente referido removemos o campo exportado da base de dados mongo uma vez que mal os dados são exportados são logo apagados, pelo que não há necessidade de os diferenciar. Alterámos também o número de Mains e consequentemente o número de threads utilizadas. Na nossa implementação, usámos apenas dois Mains: um que transporta os dados dos sensores até à base de dados mongo e outro que exporta desta para o relacional. Com isto, alterámos também a periodicidade de exportação para o relacional, que neste momento ocorre com a frequência que o administrador da aplicação defina na base de dados, isto é, o administrador da aplicação pode decidir que valor colocar no campo 'tempoExportacao' da tabela sistema, fazendo com que seja essa a periodicidade com que os dados são exportados da base de dados mongo para o relacional.

Acrescentámos à tabela de alertas o campo 'descricao' que explicita em linguagem natural o que se sucedeu para despoletar o alerta. Adicionámos ainda ao mongo os campos 'causaTemperatura' e 'causaLuminosidade' que correspondem às descrições dos alertas anteriormente explicadas.

1.8.2 Código Mongo Implementado (dentro do java)

Código responsável por ler dados da base de dados mongo:

```
/**
 * Allows to read data from a mongo database.
 * @return arraylist of read DBObjects
 * @throws InterruptedException if interrupted while waiting to read again
 */
public synchronized ArrayList<DBObject> read() throws InterruptedException {
    wait(15000);

    ArrayList<DBObject> objects = new ArrayList<>();

    MongoClient mongoClient1 = new MongoClient("localhost", 27017);
    DB db = mongoClient1.getDB("sensores_grupo10");
    DBCollection table = db.getCollection("medicoes");
    DBCursor cursor = table.find();
    while (cursor.hasNext()) {
        objects.add(cursor.next());
    }

    return objects;
}
```

Código utilizado para escrever dados na base de dados mongo

```
/**
 * Allows you to write data to a mongo database.
 * @param m is the measurement that must be recorded in the database.
 */
public void write(Medicao m) {
    MongoClient mongoClient1 = new MongoClient("localhost", 27017);
    @SuppressWarnings("deprecation")
    DB db = mongoClient1.getDB("sensores_grupo10");
    DBCollection table = db.getCollection("medicoes");

    BasicDBObject document = new BasicDBObject();
    document.put("timestamp", m.getTimestamp());
    if(m.getTemperatura() != null)
        document.put("temperatura", m.getTemperatura() + "");
    if(m.getLuminosidade() != null)
        document.put("luminosidade", m.getLuminosidade() + "");
    if(m.isAlertaTemperatura() == 1)
        document.put("alertaTemperatura", 1 + "");
    if(m.isAlertaLuminosidade() == 1)
        document.put("alertaLuminosidade", 1 + "");
    if(m.isErroTemperatura() == 1)
        document.put("erroTemperatura", 1 + "");
    if(m.isErroLuminosidade() == 1)
        document.put("erroLuminosidade", 1 + "");
    if(!m.getCausaTemperatura().equals(""))
        document.put("causaTemperatura", m.getCausaTemperatura());
    if(!m.getCausaLuminosidade().equals(""))
        document.put("causaLuminosidade", m.getCausaLuminosidade());
    document.put("exportado", 0 + "");

    try { table.insert(document);
    } catch (Exception e) {
        e.printStackTrace();
    }
    mongoClient1.close();
}
```

Código responsável por apagar do mongo as linhas que já foram exportadas

```
/**
 * Deletes all the mongoDB values from the sensores_grupo10 database.
 */
public void deleteAll() {
    MongoClient mongoClient1 = new MongoClient("localhost", 27017);
    DB db = mongoClient1.getDB("sensores_grupo10");
    DBCollection table = db.getCollection("medicoes");
    table.remove(new BasicDBObject());
}
```

1.8.3 Código SQL Implementado

Trigger implementado que verifica se os dados inseridos na tabela de medições correspondem a um alerta, para serem desta forma inseridos na tabela de alertas.

Nome do trigger: InsertMedicoes

Tipo do trigger: After Insert

BEGIN

```
DECLARE limiteS DECIMAL(8,2);
```

```
DECLARE limiteI DECIMAL(8,2);
```

```
DECLARE percentagem DECIMAL(8,2);
```

```
DECLARE nomeVariavel Varchar(100);
```

```
DECLARE nomeCultura Varchar(100);
```

```
DECLARE intervaloMed DECIMAL(8,2);
```

```
DECLARE margem DECIMAL(8,2);
```

```
SELECT variaveis_medidas.LimiteSuperior FROM  
variaveis_medidas WHERE  
variaveis_medidas.IdVariaveisMedidas =  
new.IdVariaveisMedidas INTO limiteS;
```

```
SELECT variaveis_medidas.LimiteInferior FROM  
variaveis_medidas WHERE  
variaveis_medidas.IdVariaveisMedidas =  
new.IdVariaveisMedidas INTO limiteI;
```

```
SELECT variaveis_medidas.MargemSegurancaVariavel FROM  
variaveis_medidas WHERE  
variaveis_medidas.IdVariaveisMedidas=  
new.IdVariaveisMedidas INTO percentagem;
```



```

        SELECT cultura.NomeCultura FROM cultura,
variaveis_medidas WHERE
variaveis_medidas.IdVariaveisMedidas =
new.IdVariaveisMedidas AND variaveis_medidas.IDCultura =
cultura.IDCultura INTO nomeCultura;

```

```

        SELECT variaveis.NomeVariavel FROM variaveis,
variaveis_medidas WHERE
variaveis_medidas.IdVariaveisMedidas =
new.IdVariaveisMedidas AND variaveis_medidas.IDVariavel =
variaveis.IDVariavel INTO nomeVariavel;

```

```

        SET intervaloMed = limiteS-LimiteI;

```

```

        SET margem = intervaloMed*percentagem;

```

```

        INSERT into logs VALUES (null, CURRENT_USER,
"medicoes", "INSERT", "Não Aplicável",
CONCAT("NumeroMedicao", ": ", new.NumeroMedicao, "
DataHoraMedicao", ": ", new.DataHoraMedicao, "
ValorMedicao", ": ", new.ValorMedicao, "
IdVariaveisMedidas", ": ", new.IdVariaveisMedidas),
NOW(),0);

```

```

        IF(new.ValorMedicao < limiteI)

```

```

        THEN INSERT into alertas VALUES(null, nomeVariavel,
nomeCultura, CURRENT_USER, NOW(), limiteI, limiteS,
new.ValorMedicao, "O valor da medicao ultrapassou o limite
inferior.");

```

```

        ELSEIF(new.ValorMedicao = limiteI)

```

```

        THEN INSERT into alertas VALUES(null, nomeVariavel,
nomeCultura, CURRENT_USER, NOW(), limiteI, limiteS,
new.ValorMedicao, "O valor da medicao atingiu o limite
inferior.");

```

```
ELSEIF(new.ValorMedicao > limiteI AND new.ValorMedicao  
<= limiteI+margem)
```

```
THEN INSERT into alertas VALUES(null, nomeVariavel,  
nomeCultura,CURRENT_USER, NOW(), limiteI, limiteS,  
new.ValorMedicao, "O valor da medicao esta proximo do  
limite inferior.");
```

```
ELSEIF(new.ValorMedicao >= limiteS-margem AND  
new.ValorMedicao < limiteS)
```

```
THEN INSERT into alertas VALUES(null, nomeVariavel,  
nomeCultura,CURRENT_USER, NOW(), limiteI, limiteS,  
new.ValorMedicao, "O valor da medicao esta proximo do  
limite superior.");
```

```
ELSEIF(new.ValorMedicao = limiteS)
```

```
THEN INSERT into alertas VALUES(null, nomeVariavel,  
nomeCultura, CURRENT_USER, NOW(), limiteI, limiteS,  
new.ValorMedicao, "O valor da medicao atingiu o limite  
superior.");
```

```
ELSEIF(new.ValorMedicao > limiteS)
```

```
THEN INSERT into alertas VALUES(null, nomeVariavel,  
nomeCultura, CURRENT_USER, NOW(), limiteI, limiteS,  
new.ValorMedicao, "O valor da medicao ultrapassou o limite  
superior.");
```

```
END IF;
```

```
END
```

1.8.4 Tempo Médio

| Hora da mensagem | Hora chegada ao Java | Hora da inserção no relacional | Tempo Total |
|-------------------------|-----------------------------|---------------------------------------|--------------------|
| 17/05/2019 20:13:47 | 17/05/2019 20:13:47 | 17/05/2019 20:14:08 | 0:00:21 |
| 17/05/2019 20:13:48 | 17/05/2019 20:13:48 | 17/05/2019 20:14:08 | 0:00:20 |
| 17/05/2019 20:13:50 | 17/05/2019 20:13:50 | 17/05/2019 20:14:08 | 0:00:18 |
| 17/05/2019 20:13:52 | 17/05/2019 20:13:52 | 17/05/2019 20:14:08 | 0:00:16 |
| 17/05/2019 20:13:54 | 17/05/2019 20:13:54 | 17/05/2019 20:14:08 | 0:00:14 |
| 17/05/2019 20:13:56 | 17/05/2019 20:13:56 | 17/05/2019 20:14:08 | 0:00:12 |
| 17/05/2019 20:13:57 | 17/05/2019 20:13:57 | 17/05/2019 20:14:08 | 0:00:11 |
| 17/05/2019 20:13:59 | 17/05/2019 20:13:59 | 17/05/2019 20:14:08 | 0:00:09 |
| 17/05/2019 20:14:01 | 17/05/2019 20:14:01 | 17/05/2019 20:14:08 | 0:00:07 |
| 17/05/2019 20:14:03 | 17/05/2019 20:14:03 | 17/05/2019 20:14:08 | 0:00:05 |
| 17/05/2019 20:14:05 | 17/05/2019 20:14:05 | 17/05/2019 20:14:08 | 0:00:03 |
| 17/05/2019 20:14:07 | 17/05/2019 20:14:07 | 17/05/2019 20:14:08 | 0:00:01 |
| 17/05/2019 20:14:08 | 17/05/2019 20:14:08 | 17/05/2019 20:14:08 | 0:00:00 |
| 17/05/2019 20:14:10 | 17/05/2019 20:14:10 | 17/05/2019 20:14:11 | 0:00:01 |
| 17/05/2019 20:14:11 | 17/05/2019 20:14:11 | 17/05/2019 20:14:11 | 0:00:00 |
| 17/05/2019 20:14:14 | 17/05/2019 20:14:14 | 17/05/2019 20:14:14 | 0:00:00 |
| 17/05/2019 20:14:16 | 17/05/2019 20:14:16 | 17/05/2019 20:14:17 | 0:00:01 |
| 17/05/2019 20:14:18 | 17/05/2019 20:14:18 | 17/05/2019 20:14:20 | 0:00:02 |

No gráfico anterior é possível verificar a hora a que as mensagens foram geradas, a hora que as mesmas chegaram ao java, a hora em que as mesmas foram inseridas no relacional e por fim na coluna mais à direita o tempo total desde que a mensagem foi gerada e a hora a que chegou ao relacional.

A diferença no tempo total das primeiras 13 mensagens ocorreu devido ao tempo que foi necessário até correrem o main responsável pela exportação. Desde a linha 13 em diante percebemos que em média os dados demoram um segundo a chegar ao relacional.

De referir que o parâmetro 'tempoExportacao' da tabela sistema estava configurado com o valor 3 segundos.

De notar que alguns dos valores do tempo total de exportação equivalem a 0 segundos, mas esse pormenor deve-se à precisão da medição associada aos testes. No pior dos casos demora 3 segundos a fazer a exportação dado que é esse o intervalo de espera das mensagens.

































Se usássemos um mecanismo de exportação baseado wait and notify , conseguíamos valores na ordem dos milissegundos, contudo estaríamos a sacrificar a robustez e versatilidade do programa (pois preferimos usar 2 mains em vez de 1).

1.8.5 Alertas














































Tendo a tabela 'Sistema' os seguintes parâmetros:

| | | | | |
|---------------------------|--------------------------------|----------------------------|-------------------------------|-------------------|
| + Opções | | | | |
| LimiteInferiorTemperatura | LimiteSuperiorTemperatura | MargemSegurancaTemperatura | LimiteInferiorLuz | LimiteSuperiorLuz |
| 20.00 | 25.00 | 0.10 | 1000.00 | 5000.00 |
| | | | | |
| MargemSegurancaLuz | PorcentagemVariacaoTemperatura | PorcentagemVariacaoLuz | TempoEntreAlertasConsecutivos | TempoExport |
| 0.10 | 0.10 | 0.10 | 10.00 | 10.00 |

E sendo estes os valores recebidos para a luminosidade:

| | | | | |
|--------------------------|---|--|--|---------------------|
| + Opções | | | | |
| ← T → | | | | |
| | | DataHoraMedicao | ValorMedicaoLuminosidade | IDMedicao |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:04 |
| | | | 730.00 | 1331 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:06 |
| | | | 730.00 | 1332 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:08 |
| | | | 730.00 | 1333 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:10 |
| | | | 730.00 | 1334 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:12 |
| | | | 730.00 | 1335 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:14 |
| | | | 731.00 | 1336 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:15 |
| | | | 731.00 | 1337 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:17 |
| | | | 731.00 | 1338 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:19 |
| | | | 730.00 | 1339 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:21 |
| | | | 731.00 | 1340 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:23 |
| | | | 730.00 | 1341 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:24 |
| | | | 731.00 | 1342 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:26 |
| | | | 731.00 | 1343 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:28 |
| | | | 731.00 | 1344 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:30 |
| | | | 731.00 | 1345 |

E sendo estes os valores recebidos para a temperatura:

| | | | | |
|--------------------------|---|--|--|---------------------|
| + Opções | | | | |
| ← T → | | | | |
| | | DataHoraMedicao | ValorMedicaoTemperatura | IDMedicao |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:04 |
| | | | 26.60 | 1335 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:06 |
| | | | 26.60 | 1336 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:08 |
| | | | 26.60 | 1337 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:10 |
| | | | 26.60 | 1338 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:12 |
| | | | 26.50 | 1339 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:14 |
| | | | 26.50 | 1340 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:15 |
| | | | 26.50 | 1341 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:17 |
| | | | 26.50 | 1342 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:19 |
| | | | 26.50 | 1343 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:21 |
| | | | 26.50 | 1344 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:23 |
| | | | 26.50 | 1345 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:24 |
| | | | 26.50 | 1346 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:26 |
| | | | 26.60 | 1347 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:28 |
| | | | 26.60 | 1348 |
| <input type="checkbox"/> |  Edita |  Copiar |  Apagar | 2019-05-19 18:34:30 |
| | | | 26.60 | 1349 |

Serão ambos considerados alertas uma vez que:

Para a luminosidade: $730 < 1000$

Para a temperatura: $26.6 > 25$

| + Opções | | | | | | | | | | |
|--------------------------|---|----------|--------------|-------------|-------------------|---------------------|-------------------|-------------------|--------|---|
| | | idAlerta | nomeVariavel | nomeCultura | emailInvestigador | data | limiteInferiorVar | limiteSuperiorVar | valor | descricaoAlertas |
| <input type="checkbox"/> | ✎ | 103 | luz | todas | todos | 2019-05-19 18:33:30 | 1000.00 | 5000.00 | 729.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ | 105 | luz | todas | todos | 2019-05-19 18:33:41 | 1000.00 | 5000.00 | 729.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ | 107 | luz | todas | todos | 2019-05-19 18:33:52 | 1000.00 | 5000.00 | 729.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ | 109 | luz | todas | todos | 2019-05-19 18:34:01 | 1000.00 | 5000.00 | 730.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ | 111 | luz | todas | todos | 2019-05-19 18:34:12 | 1000.00 | 5000.00 | 730.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ | 104 | temperatura | todas | todos | 2019-05-19 18:33:30 | 20.00 | 25.00 | 26.60 | O valor da medicao da temperatura ultrapassou o L... |
| <input type="checkbox"/> | ✎ | 106 | temperatura | todas | todos | 2019-05-19 18:33:41 | 20.00 | 25.00 | 26.60 | O valor da medicao da temperatura ultrapassou o L... |
| <input type="checkbox"/> | ✎ | 108 | temperatura | todas | todos | 2019-05-19 18:33:52 | 20.00 | 25.00 | 26.60 | O valor da medicao da temperatura ultrapassou o L... |
| <input type="checkbox"/> | ✎ | 110 | temperatura | todas | todos | 2019-05-19 18:34:01 | 20.00 | 25.00 | 26.60 | O valor da medicao da temperatura ultrapassou o L... |
| <input type="checkbox"/> | ✎ | 112 | temperatura | todas | todos | 2019-05-19 18:34:12 | 20.00 | 25.00 | 26.50 | O valor da medicao da temperatura ultrapassou o L... |

Pode-se verificar que os alertas estão a ser recebidos de 10 em 10 segundos como especificado no sistema.

Alterando a tabela sistema para :

| + Opções | | | | | |
|---------------------------|---------------------------|--------------------------------|------------------------|-------------------------------|-------------|
| LimiteInferiorTemperatura | LimiteSuperiorTemperatura | MargemSegurancaTemperatura | LimiteInferiorLuz | LimiteSuperiorLuz | |
| 20.00 | 27.00 | 0.10 | 700.00 | 740.00 | |
| | | | | | |
| LimiteSuperiorLuz | MargemSegurancaLuz | PercentagemVariacaoTemperatura | PercentagemVariacaoLuz | TempoEntreAlertasConsecutivos | TempoExport |
| 740.00 | 0.10 | 0.10 | 0.10 | 10.00 | 3.00 |

Os alertas serão estes para os mesmos valores de temperatura e luminosidade recebidos:

| + Opções | | | | | | | | | | |
|--------------------------|---|----------|--------------|-------------|-------------------|---------------------|-------------------|-------------------|--------|---|
| | | idAlerta | nomeVariavel | nomeCultura | emailInvestigador | data | limiteInferiorVar | limiteSuperiorVar | valor | descricaoAlertas |
| <input type="checkbox"/> | ✎ | 117 | luz | todas | todos | 2019-05-19 18:42:50 | 700.00 | 740.00 | 693.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ | 119 | luz | todas | todos | 2019-05-19 18:43:02 | 700.00 | 740.00 | 688.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ | 116 | temperatura | todas | todos | 2019-05-19 18:42:37 | 20.00 | 27.00 | 26.60 | O valor da medicao da temperatura esta proximo do ... |
| <input type="checkbox"/> | ✎ | 118 | temperatura | todas | todos | 2019-05-19 18:42:50 | 20.00 | 27.00 | 26.60 | O valor da medicao da temperatura esta proximo do ... |
| <input type="checkbox"/> | ✎ | 120 | temperatura | todas | todos | 2019-05-19 18:43:02 | 20.00 | 27.00 | 26.60 | O valor da medicao da temperatura esta proximo do ... |

Uma vez que ainda não foram ultrapassados os limites definidos, mas encontra-se na margem de segurança definida, o alerta serve de precaução.

Já para as medições feitas manualmente, verifica-se o mesmo raciocínio, implementado com recurso a triggers.

Na tabela variáveis medidas:

| + Opções | | IDVariavel | IDCultura | LimiteInferior | LimiteSuperior | MargemSegurancaVariavel | IdVariaveisMedidas |
|--------------------------|----------------------------|------------|-----------|----------------|----------------|-------------------------|--------------------|
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 3 | 4 | 2.85 | 8.25 | 0.10 | 1 |
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 3 | 7 | 3.00 | 4.00 | 0.80 | 2 |
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 3 | 8 | 4.00 | 6.00 | 0.10 | 3 |
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 3 | 8 | 4.00 | 5.00 | 0.10 | 4 |

↑ ☐ Marcar todos Com os seleccionados: ✎ Editar ✂ Copiar ✖ Apagar 📄 Exportar

Na tabela de alertas:

| + Opções | | idAlerta | nomeVariavel | nomeCultura | emailInvestigador | data | limiteInferiorVar | limiteSuperiorVar | valor | descricaoAlertas |
|--------------------------|----------------------------|----------|--------------|-------------|-------------------|---------------------|-------------------|-------------------|--------|---|
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 105 | luz | todas | todos | 2019-05-19 18:33:41 | 1000.00 | 5000.00 | 729.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 117 | luz | todas | todos | 2019-05-19 18:42:50 | 700.00 | 740.00 | 693.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 119 | luz | todas | todos | 2019-05-19 18:43:02 | 700.00 | 740.00 | 688.00 | O valor da medicao da luminosidade esta abaixo do ... |
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 116 | temperatura | todas | todos | 2019-05-19 18:42:37 | 20.00 | 27.00 | 26.60 | O valor da medicao da temperatura esta proximo do ... |
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 118 | temperatura | todas | todos | 2019-05-19 18:42:50 | 20.00 | 27.00 | 26.60 | O valor da medicao da temperatura esta proximo do ... |
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 120 | temperatura | todas | todos | 2019-05-19 18:43:02 | 20.00 | 27.00 | 26.60 | O valor da medicao da temperatura esta proximo do ... |
| <input type="checkbox"/> | ✎ Editar ✂ Copiar ✖ Apagar | 121 | Chumbo | bananas | root@localhost | 2019-05-19 18:47:58 | 4.00 | 5.00 | 3.00 | O valor da medicao ultrapassou o limite inferior. |

↑ ☐ Marcar todos Com os seleccionados: ✎ Editar ✂ Copiar ✖ Apagar 📄 Exportar

2 Android e Php

2.1 Esquema da BD Lite Geral

Não Aplicável uma vez que o esquema da base de dados do Android é igual para todos os grupos.

2.2 Layout Implementado no Android

