Budapesti Műszaki és Gazdaságtudományi Egyetem
Villamosmérnöki és Informatikai Kar

Hőmérsékleti szenzorok adatainak vizualizációja és monitorozása

2018

TARTALOMJEGYZÉK

1. Általános tudnivalók

Az alábbi linkeken találhatóak információk a laboron használt technológiákról. A labort kezdjük az alábbi linkek átfutásával.

1.1. Grafana

http://docs.grafana.org/guides/basic_concepts/

http://docs.grafana.org/features/panels/graph/

http://docs.grafana.org/alerting/notifications/

http://docs.grafana.org/alerting/notifications/#webhook

http://docs.grafana.org/alerting/rules/

http://docs.grafana.org/reference/timerange/

http://docs.grafana.org/features/datasources/influxdb/

1.2. InfluxDB

https://docs.influxdata.com/influxdb/v1.7/concepts/key_concepts/

https://docs.influxdata.com/influxdb/v1.7/query_language/data_exploration/

https://docs.influxdata.com/influxdb/v1.7/query_language/data_exploration/#select-data-that-have-a-specific-tag-key-value

https://docs.influxdata.com/influxdb/v1.7/query_language/database_management/

https://docs.influxdata.com/influxdb/v1.7/query_language/functions#moving-average

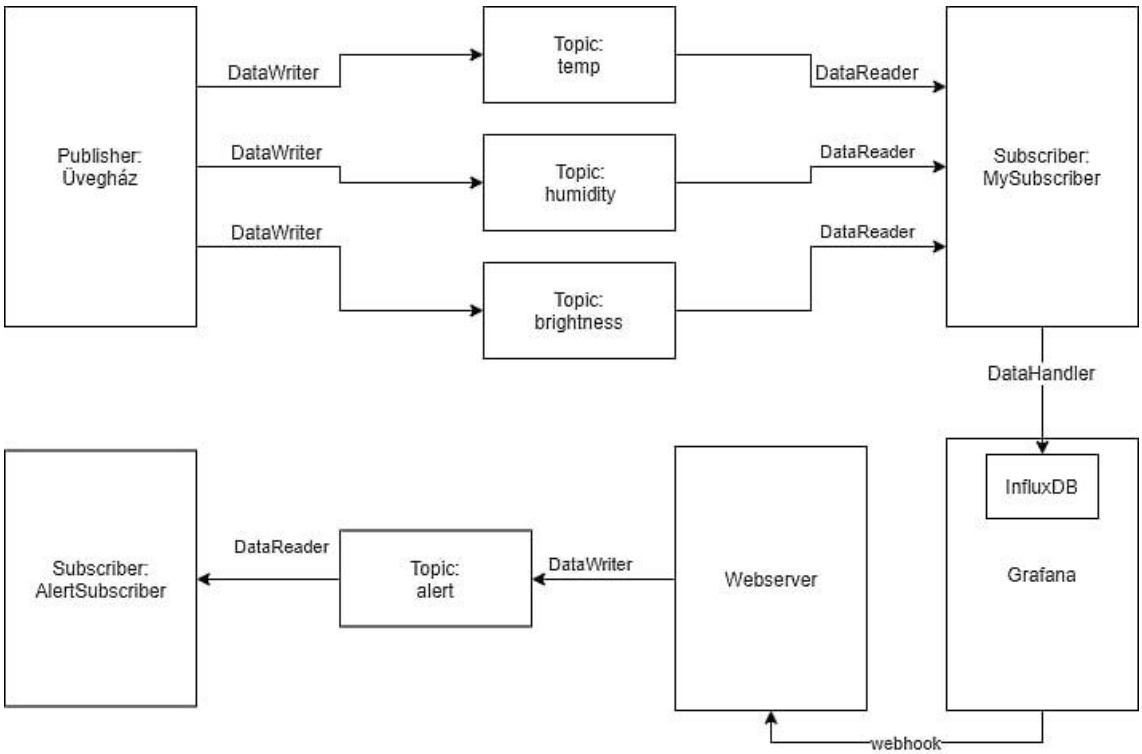
2. Szoftverkörnyezet

Az alábbi szoftverkörnyezetet fogjuk használni a laboron. Ezen szoftverek közül több is megjelent már a korábbi tanulmányok részeként.

- *Eclipse*
- *RTI Connext*: DDS implementáció
- *Grafana*: Egy adatvizualizációra és monitorozásra használt eszköz. Különböző adatforrásokat lehet hozzáadni és dashboardokat lehet benne létrehozni a különböző grafikonoknak. Illetve riasztások és értesítési csatornák segítségével jelezni tudja, ha egy megjelenítőn valami nem kívánt anomália lépett fel vagy szűnt meg.
- *InfluxDB*: Egy idősorok tárolására használt adatbáziskezelő rendszer, amely sokban hasonlít a közkedvelt MySQL-re.

3. Szenzor adatok elérése

Az alábbi ábra szemlélteti a használt példarendszer logikai/kommunikációs felépítését.



A szenzorból adatokat a DDS segítségével nyerhetünk. Jelen labor alatt egy előre felvett mérés visszajátszását fogjuk végrehajtani, amellyel azt szimuláljuk, mintha az adatokat valós időben kapnánk a rendszerből. Az adatok egy üvegház viselkedését fogják szimulálni, ahonnan hőmérséklet, páratartalom és fény adatok érkeznek.

A példaként kiadott Eclipse projektben a teljes folyamat el van készítve, a DDS kommunikáció, az adatok InfluxDB-be történő kiírása.

Az InfluxDB a cps adatbázisba dolgozik, azon belül található egy cps nevű measurement (analógia a relációs adatbázisban tábla). Van egy tag (oszlopok, amelyek mindig karakterláncok, ezekre épít indexet az adatbáziskezelő rendszer, így ezekre érdemes keresni) értéke, aminek a neve name. És egy field (erre csak lineáris kereséssel tudunk keresni) valós szám, ez reprezentálja a hőmérsékletet, a neve pedig value.

Példa adatbázis

cps,name=temp0 value=27.9 <TIMESTAMP>

cps,name=temp0 value=27.9 <TIMESTAMP>

cps,name=temp0 value=27.9 <TIMESTAMP>

cps,name=temp1 value=27.9 <TIMESTAMP>

cps,name=temp2 value=27.9 <TIMESTAMP>

Példa lekérdezés: SELECT “value” FROM “cps” WHERE “name”=’temp2’

4. Vizualizáció

Ebben a fejezetben gyorsan áttekintjük a gyakorlat kezdő lépéseit.

A virtuális gép elindítása után a grafana és az influxdb szolgáltatás automatikusan elindul: érdemes ellenőrizni, amit a http://localhost:3000 webcímen böngészőből tehetünk meg. A továbbiakban ezt fogjuk használni. Az alapértelmezett bejelentkezési adatok: admin/admin.

Indítsa el a Grafana szervert és lépjen be a böngészőben a kliens oldalra. Az adatforrást előzetesen bekonfiguráltunk, hogy az InfluxDB legyen. Egy példa Dashboard is elérhető, amibe már be van állítva, hogy az InfluxDB-ből olvassa be a szenzor adatait. Ezt szemrevételezzük!

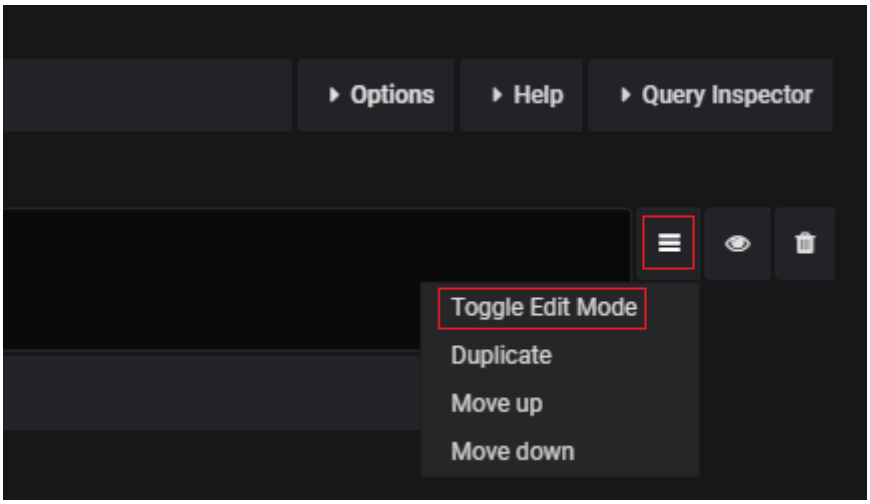
Feladat

Nyissa meg az Eclipse projektet és módosítsa úgy, hogy miután a Subscriber megkapta az adatot, ne csak akkor küldje tovább az InfluxDB-nek, ha a küldő szenzor azonosítója temp0, hanem akkor is, ha **temp1** vagy **temp2**, ezzel három szenzor adatait fogjuk egyszerre megkapni. A gyakorlat során több, mint 10 különböző szenzor adatot fogaduk, amelyből mindenki ki tudja választani a neki tetsző adatokat. Az utolsó két sorszámú szenzor a külső hőmérséklet.

A Grafana felületen hozzon létre egy új Dashboardot vagy használja a példát, adjon hozzá 6 darab panelt (Graph panelek legyenek). Állítsa be, hogy az adatforrásuk az InfluxDB legyen.

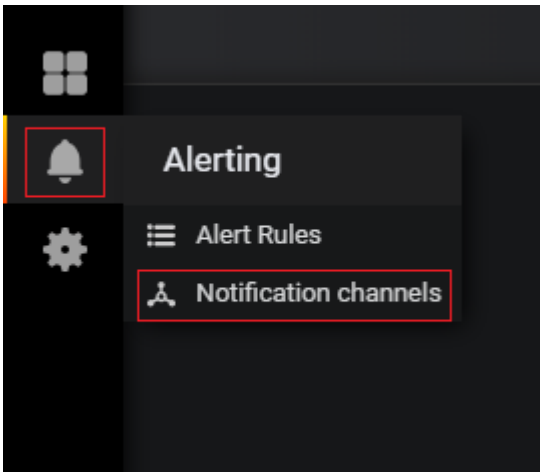
Ha új Dashboardot hozott létre, állítsa be, hogy a figyelt intervallum folyamatosan frissüljön.

Külön-külön mind a háromra állítsa be, hogy az az egyes szenzorok adatait mutassák (tehát panel1 és panel2 akkor, ha pl. temp0 az azonosító, panel3 és panel4 akkor, ha pl. temp1 az azonosító). Érdemes az Influx lekérdezést leírni/megírni az összekattintgatás helyett.



5. Monitorozás

A Grafana felületen meg lehet adni, hogy az egyes *Alert*eket hova küldje el a szerver. A labor környezetben már előre bekonfiguráltunk egy webhookot, melynek lényege, hogy egy POST vagy PUT kérést küld egy adott webcímre (jelen esetben ez a http://localhost:6354 és POST kérés, a webszerver, amely kiszolgálja ezt a kérést megtalálható a kiadott kódban „*ServerMain.java*” néven. Ennek indításával fogadhatjuk az *Alert*eket és DDS-en keresztül ki is tudjuk őket küldeni).



A Grafana hátránya, hogy csupán egy Alertet lehet létrehozni Panelenként és az ÉS/VAGY feltételeket sem szereti igazán, alapvetően nem a komplex monitorozás támogatására hozták létre. Érdeklődés esetén érdemes megnézni a *Prometheus* monitorozó keretrendszert.

Feladat 1

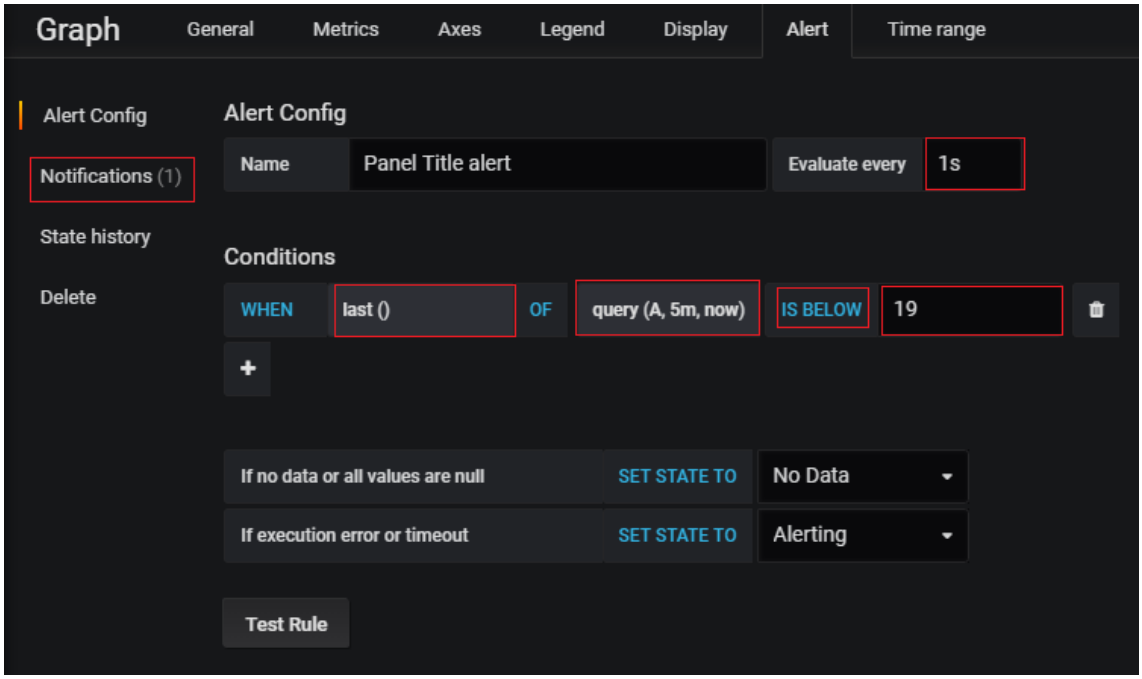
A példarendszerünkben riasztást szeretnénk kiadni, ha a szenzorok által mért értékek túl magasak vagy túl alacsonyak. Ez egy valósághoz közeli szcenárió.

Hozzon létre kettő-kettő új riasztást minden panelhez az alábbi feltételekkel! Adja meg, hogy küldjön riasztást a rendszer, ha:

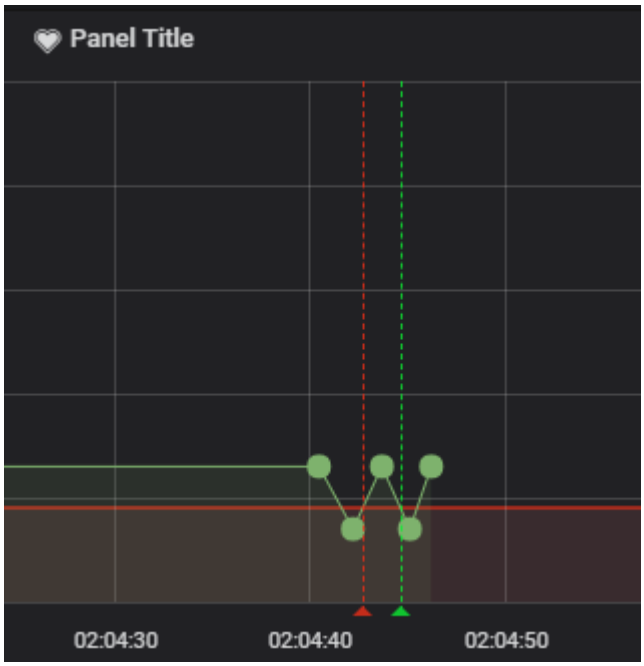
- A lekérdezés értékének utolsó eleme kisebb, mint 19 (panel1, panel3, panel5)
- A lekérdezés értékének utolsó eleme nagyobb, mint 25 (panel2, panel4, panel6)

Ezzel nagyjából azt tudjuk szimulálni, hogy riasztást adunk ki, ha az üvegházban túl alacsony vagy túl magas értékeket mérnek a szenzorok. Egyik sem előnyös ugyanis.

Állítsa be, hogy a Notification channel az előre bekonfigurált webhook legyen. A notification üzenete a neve legyen, lehetőleg ékezetek nélkül és valami azonosító pl. panelX_alert mondjuk szóközzel elválasztva a kettőt.



A lenti ábra vízszintes piros sávjában láthatjuk a riasztás kritikus sávját, ha bármelyik érték belép oda, akkor egy critical alert jön létre, ezt reprezentálja a szaggatott piros függőleges vonal, a zöld pedig, hogy az adat kilépett a kritikus állapotból és újra normál értéket vesz fel.



Feladat 2

Az alábbiakban kipróbáljuk az egyszerűbb számítási és riasztási funkciókat.

- Adjon hozzá két új Graph panelt a Dashboardhoz.
- A hozzájuk írt lekérdezések kérjék le mindenféle szűrés nélkül a *value előző 5 értékének mozgóátlagát*.

Állítsa be a következő riasztásokat:

- A lekérdezés értékének utolsó eleme kisebb, mint 19 (panel7)
- A lekérdezés értékének utolsó eleme nagyobb, mint 25 (panel8)

Ezzel nagyjából azt tudjuk szimulálni, hogy riasztást adunk ki, ha az üvegházban a hőmérséklet nem felel meg az általunk elvártaknak.

Ugyanezen feladatokat csinálja végig egy másik típusú mérésre is!