Alternatives considered: splitting databases with different functionalities. However, this seemd inpratical and it much more sense to ‘collect’ all data about devices in one databses and let different components interact with it through a common interface.

**AssignmentDB, ConfigurationDB, PeripheralTypeDB … 🡪 MainDB**

MainDB: stores users, delivery methods, notifications itself (for later consultation) …

TODO: Dashboards/façades for different end users ?

**M1 : Integrate new sensor or actuator manufacturer**

* new types of sensor or actuator data should be transmitted, processed and stored, and made available to applications

**DataFlowManager (not different than other sensors), SensorDataManager, SensorDataDB, ApplicationFacade**

* data processing subsystem should be extended with relevant data conversions

**Conversion methods are stored in MainDB, DataFlowManager consults conversion methods**

* available applications can be updated to use any new pluggable devices.

No need for new structures, there is no distinction between new sensors and already existing sensors

* infrastructure managers must be able to initialize the new type of pluggable device, configure access rights for these devices, and view detailed information about the new type of pluggable device

**Infrastructure owner can use the DeviceMgmt for these actions: changes in the configurations can result in applications becoming (non-) active and this is the responsibility of the DataManager. Topology can be shown using the RequestTopology interface. Consulting log files is also done using this interface: logs are generated when connection/disconnection/heartbeats are transmitted through the system in the MainDB.**

* + UC8 : Initialise a pluggable device
  + UC9 : Configure pluggable device access rights
  + UC10 : Consult and configure the topology

**U2: Easy installation**

* gateway should not require any configuration, other than being connected to the local wired or WiFi network, after it is plugged into an electrical socket.

**Gateways transmit a connection message we assume that the connection message emits a working default configuration.**

* Installing a new mote should not require more configuration than adding it to the topology
  + UC4: Install mote

**We assume this broadcast message is the same as a connection message. The notification is sent via the NotificationHandler to the infrastructure owner.**

* Reintroducing a previously known mote, with the same pluggable devices attached to it, should not require any configuration.

**The DataManager checks this in the MainDB. We keep track of previously connected devices simply by toggling their ‘activation status’ in the MainDB. Whenever it’s reconnected, the DataManager can restore the status to active (the same for the motes that were connected to it) and nothing more is required.**

* Adding new sensors or actuators should require no further customer actions besides plugging it into the mote. Configurable sensors and actuators should have a working default configuration.

**Again, this is the responsibility of the DataManager (restoring configuration, restoring acces rights). It keeps track of Applications waiting for these pluggables or Applications previously using these sensors and notifies them of this device change, which can result in these applications becoming active. DataManager is responsible for notifying.**

* + UC6 : Insert a pluggable device into a mote
* Applications should work out of the box if the required sensors and actuators are available.

**ApplicationFacade can request the topology and activate the Application if the required sensors are available, active and configured in the Main DB. This is done via the ApplicationActivator. Lastly, we assume roles can not be changed separately after configuring them; put simple: changing roles is can not be done separate from subscribing to an application and can thus not result in an application becoming inactive.**

* Only when mandatory end-user roles must be assigned, additional explicit configuration actions are requirement from a customer organisation (cf. UC17 : Activate an application and UC19 : Subscribe to application).

**Subscriber etc. (see later)**

**AV3: Pluggable device or mote failure**

Prevention: redundant sensors

* Application providers can register redundant sensors in ApplicationActivationDB, next to the regular sensors (possible tactic: ‘predictive model’), thus customer organisations subscribing to applications will have to provide multiple (redundant) sensors.

In ApplicationFacade, ApplicationActivator provides the interface SensorMgmt to ApplicationDataManager. This is used to determine which is the current sensor used for data if there are redundant sensors. Whenever all sensors are unavailable the applications is set to inactive an no useful data is transmitted anymore. In short: application activation status are contained in the application itself.

Detection: heartbeats

* Responsibility of the gateway. Is forwarded via the HeartbeatHandler to notify the system of a heartbeat change an take necessary actions (mark applications as inactive, deactivate connected sensors …). This is done via the MainDB.

Resolution:

* The DataManager is responsible for notifying the infrastructure owner via the NotificationHandler.
* Idem
* Notifying the CustomerOrganisation of activation, reactivation or suspension of applications is handled by the ApplicationFacade, it uses the NotificationHandler to perform this action.
* When a device change is coming through the system the DataManager finds out which application use the devices to transmit the device change (possibly resulting in the suspension of the application) and notifies this application via the notificationhandler.

**Av1: Communication between SIoTIP gateway and Online Service**

* Detection of gateway not sending data anymore is done by SynchronizationHandler in the GatewayFacade. It timer in DeadlineChecker is not refreshed timely, the system administrator will be notified.
* Acknowledging each message is done by the handlers in the façade. If one of the acks does not arrive, the gateway starts storing its data locally. **(FIFO, LIFO … TODO: Explain !)**
* The GatewayFacade can notify SysAdmin in case of gateway communcation failure AND Monitor pings the GatewayFacade and notifies the SysAdmin in case of an internal gatewayfacade failure.

**Av2: Application failure**

* Ping tactic with monitor in Application Façade which notifies SysAdmin and after 3 attempted restarts notifies the customer organization and the application provider.

**P2: Requests to the pluggable data database**

* SensorDataManager is responsible for FIFO, aging, dynamic priority scheduling and whatnot

**U1: Application updates**

* Updating applications happens via the ApplicationUpdater.

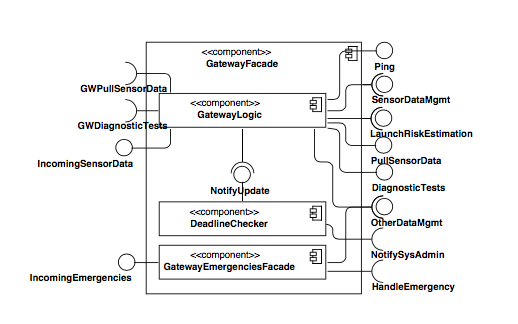
FUTURE:

UC19: Subscribe to application

* Component subscriber laten communiceren met Customer Organization die de handling van subscriptions doet.

QUESTIONS:

* Why sometimes bollekes and sometimes not?



* How do you check internal failures autonomously? Do we need to use an internal monitor, what if the monitor fails?