Key Decisions:

Added *CommunicationComponentMonitor* in GW and in OS to monitor failure of their individual communication components.

OS

-> GatewayResponseChecker - controls synchranization interval of gateway to detect failure of communication chanel / Gateway

-> GatewayMonitor - sends ping to detects if Gateway is working

Gateway

-> OSMessageChecker - checks acknowledgements from OS to detect failure

For communication between Online Service and gateway were created communication components. The OSFacade in Gateway and the GatewayFacade in Online Service. To make easier life for programers and try to avoid partial failures, we decided to use Broker pattern for communication. The communication concerns we separated from the application concerns.

We hide the implementation details of remote service invocation by encapsulating them into a layer other than the business component itself. This layer provides an interface to the Online Service that allows it to invoke methods just as it would invoke any local interface.

GatewayFacade
*interface Communication

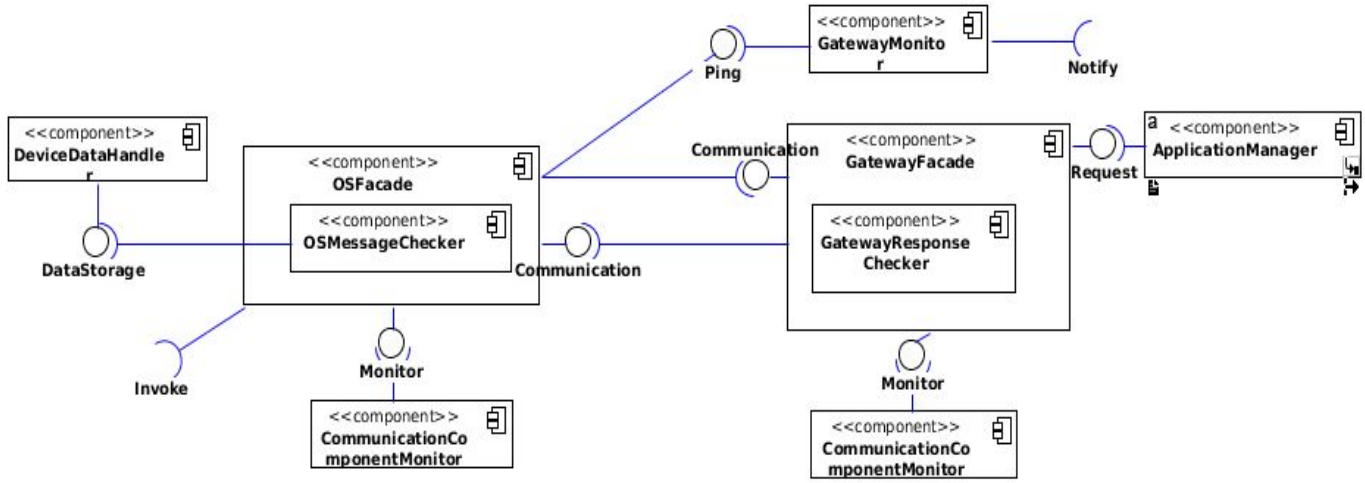
deactivateApplication(...)

sendData(Object remoteObject)
receiveData()

OSFacade
*interface Communication

sendData(Object remoteObject)
receiveData

Between Gateway and OS are Communication interfaces



Communication channel failure

The communication channel between the SIoTIP gateway and the Online Service has failed. The Online Service is able to detect that a SIoTIP gateway is not sending data anymore based on the expected synchronisation interval. An outage is defined as 3 consecutive expected synchronisations that do not arrive within 1 minute of their expected arrival time.In SIoTIP is designed *GatewayResponseChecker* component that control expected arrival time for data from Gateway and also count number of synchronization.

The Online Service should acknowledge each message sent by the SIoTIP gateway so that the gateway can detect failures. The *OSMessageChecker* checks if all acknowledges were received from Online Service within 30 sec, because he failure of an internal SIoTIP Online Service component is detected within 30 seconds.

If the communication channel has failed, the *DeviceDataHandler* on gateway will temporarily store all incoming pluggable data and any issued application commands internally. The SIoTIP gateway can store at least 3 days of pluggable data before old data has to be overwritten. If the Online Service becomes unreachable, application parts running locally on the SIoTIP gateway continue to operate normally.

The SIoTIP gateway (or an internal communication component) has failed.

The *GatewayMonitor* in Online Service was created for detection of failure of gateway. This component sends Ping regularly and notifies the infrastructure manager(within 5 minutes after the detection) when the outage of a SIoTIP gateway is detected. This is important because SIoTIP gateway, due to the occurred failure, cannot contact a system administrator itself. A SIoTIP system administrator should be notified within 1 minute after the detection of a simultaneous outage of more than 1% of the registered gateways. -> The GatewayMonitor should change status of Gateway in DeviceDB. And also call method to find out how many gateways are failed.

The SIoTIP gateway is able to autonomously detect failures of its individual internal communication components. The *CommunicationComponentMonitor* on GW send monitor GW every 30sec.
If an OSFacade fails, the *CommunicationComponentMonitor* first tries to restart the afected OSFacade component. If the failure persists, the SIoTIP gateway reboots itself entirely..

The Online Service (or an internal communication component) has failed.
The Online Service is able to autonomously detect failures of its individual internal communication components.. The failure of an internal SIoTIP Online Service component is detected within 30 seconds. CommunicationComponentMonitor in OS monitors OnlineService every 30sec.