Open Architectures for Intelligent Solid State Lighting Systems

OpenAIS IoT Lighting Controls: How (and why) it works

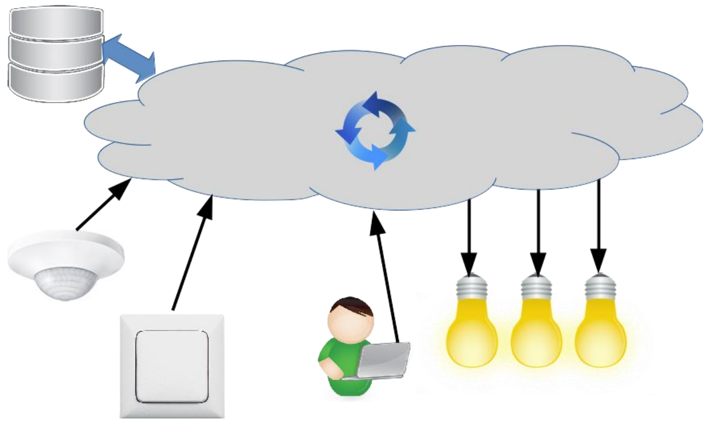
Dr. Walter Werner
on behalf of Zumtobel and Tridonic

Supported by the Horizon 2020
funding of the European Union

//Table of Content

//A Double Goal for a Single Application

- IoT tends to be a synonym for „Things get connected to clouds or central servers, that operate on their data and control their actions“.



- A) Future lighting is **controlled by „the cloud“** and users interact with „the cloud“ to control their lights. („The cloud“ may be a central server also)
- B) Lighting needs a **fast, local, and resilient operation** mode, that allows users to control their lights reliably.

//Choosing an IoT Framework for Goal 1

- Select something that works (= somehow mature and available)
- Select something that is sufficiently flexible to cover advanced and professional lighting control needs.
- Prefer RESTful and CoAP over other technologies.
- Try to estimate what might be used as a future standard in IoT.

- **We chose LWM2M**, today OCF / open connectivity / iotivity would possibly also work.
- Today there are more frameworks available, specifically from cloud vendors, most of them based on (the event - based) MQTT instead of the RESTful CoAP protocols.

//Analyzing the Needs for Goal 2

- Fast: 200ms from action to reaction
- Secure: Only authorized access to lighting
- Privacy: Status information protected from sniffing
- Sensor information linked to multiple areas
- Light Point Grouping
- Sensor Grouping
- Grouping of Groups
- Reliable: Commissioned „wiring“ of groups.
- Not in conflict with goal 1 IoT framework.
- Allows (some) local user control.

//Designing the OpenGroupCommunication

- IPv6 multicast provides a fast and reliable node-to-node communication.
- Multicast communication needs to be enhanced by serial unicast, as some nodes may not be reachable by multicast messages.
- A lighting node may host multiple light point object instances and multiple sensors, therefore a group message must be addressable to specific object types and instances of a node.
- Fast group communication security can be provided using symmetric keys.
- All group communication to be encrypted

//Designing the OGC (cont.)

- Use CoAP and a URI structure close to LWM2M for convienience.
- Use CBOR object representation to save bandwidth.
- Use OSCORE as an object security layer on top of CBOR.
- Replace ACK messages by regular but randomized status reports.
  - ACK for large multicast groups are prone to create jamming traffic.
  - Status messages provide more information than ACK, can be used to correct settings that were lost by missing group messages; the bandwidth load can be adjusted by the interval setting.

//OGC Structure of Grouping:

- Group communication is structured by „Application Group IDs“.
  - Actuators (objects) are members of application groups,
  - Controllers control application groups (and their members of course), and
  - Sensors send status messages to Application groups. (Subscription simply happens by listening to the application group, sensors do not need to maintain a subscription list)
  - Application groups use IPv6 multicast (and in addition serial unicast) addresses to address all nodes that host grouped objects.
  - Application groups share security objects that allow the protection of group messaging.
- The Application Group ID becomes part of the URI, and is replaced at the receiving node by the appropriate object instance IDs.

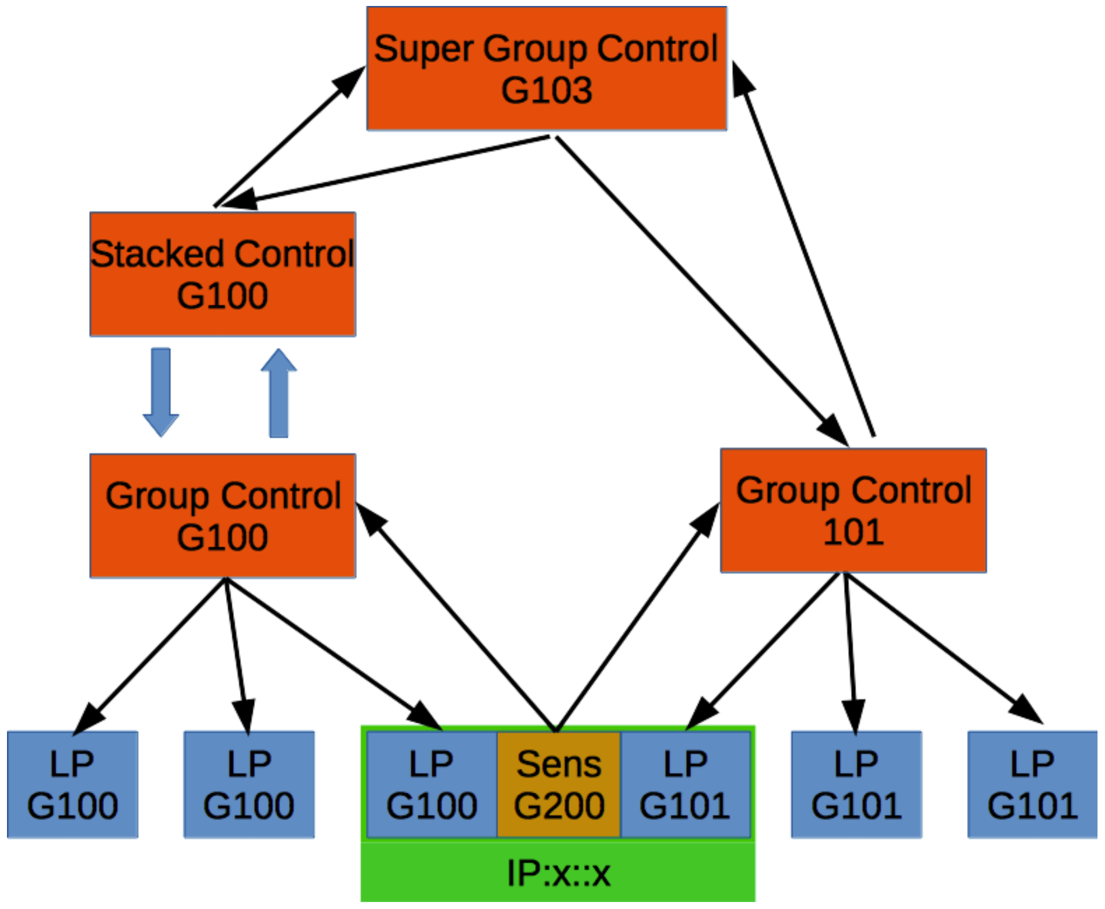
//Addressing objects & resources in OGC:

- Think of a call using the structure: coap://multicast-IP/Object-ID/Instance-ID/Resource-ID{payload}
  - Trouble is: The Instance-ID is not shareable across the nodes of a group, and there may be more than one instance at a node member of a group.
- OGC uses the Application-Group ID instead of the instance- ID to address the resources.
- The group-Object for a specific application group of the receiving nodes has a „members“ section that provides the pointers to the correct instances related to this group. This allows to have more than one instance to be part of a group.
- Working with group objects to configure group memberships allows for a fully commissioned and stable „wiring“ of groups.
- See all the details on URI structures and implementation hints in the public architecture document on www.openais.eu/en/results/

//Create Lighting Controls using OGC

- Sensors send out their (changing) status to their sensor group (= application group)
- Control objects listen to the sensor status messages.
- Control objects set actuators in their group to output values accordingly. They use grouped messages for uniform and immediate action (including scene recall) and single addressed messages for detailed adjustment (including the individual preparation of scenes)
- Scenes are controller defined and actuator cached: The caching is used as an execution optimization to achieve a decent timing.
- Control Objects provide a User API, that allows user APPs to operate the group as a whole (using the fast and reliable OGC mechanisms provided by the control object without the need to get clearance for a direct communication to each node) and also adjust single actuators.

//How (local) Groups Control Interacts



//Seamless Control Integration

- Make the group control object „**pure software**“ that allows for „**any placement**“, including luminaires, sensors router, server or „cloud“ devices.
- Make the control object **stackable**, so that a local group control may be used as „local executing representative“ of a cloud based service (as long as the connection is decently available)
- Same stacking may be used to have a control object in „shadow“ (or ghost) mode in parallel to an operative one, and have it taking control automatically when the operating control object gets disconnected or breaks.
- Make the group **control object accept actuator** controls (of a super-group) to build a group control hierarchy without limits.
- See further concepts and more details on the **public architecture document** on www.openais.eu/en/results

//Data Collection in Restricted Networks

- OGC provides a very bandwidth effective method to continuously monitor all the (always changing) status of a system:
  - The actuators and sensors provide their status messages to the control objects using OGC.
  - A „data collection“ service simply listens to this data stream by also listening to the multicast addresses and will this way be able to collect and provide to cloud-type (or BMS) services all status without ANY additional bandwidth usage.
  - This is especially useful in large networks with a lot of objects (as lighting typically has) that utilize bandwidth limited RF meshed networks.

//OGC boosts IoT Lighting Controls

- OGC provides a fast and reliable way to handle groups.
- Embedded group control objects act as „performance boosters“ for the cloud based supervising controls, and provide
  - fast and reliable handling of manual input
  - secure and fast enabler for APP User Interfaces.
- OGC provides an addition to a standard „connected RESTfully to the cloud“ setting
  - enabling resilience by utilizing embedded control
  - enabling scalability by optimizing bandwidth use
  - enabling restricted devices by handling group access policies at a single point.
- OGC makes complex and seamless grouping of sensors and actuators simple and reliable

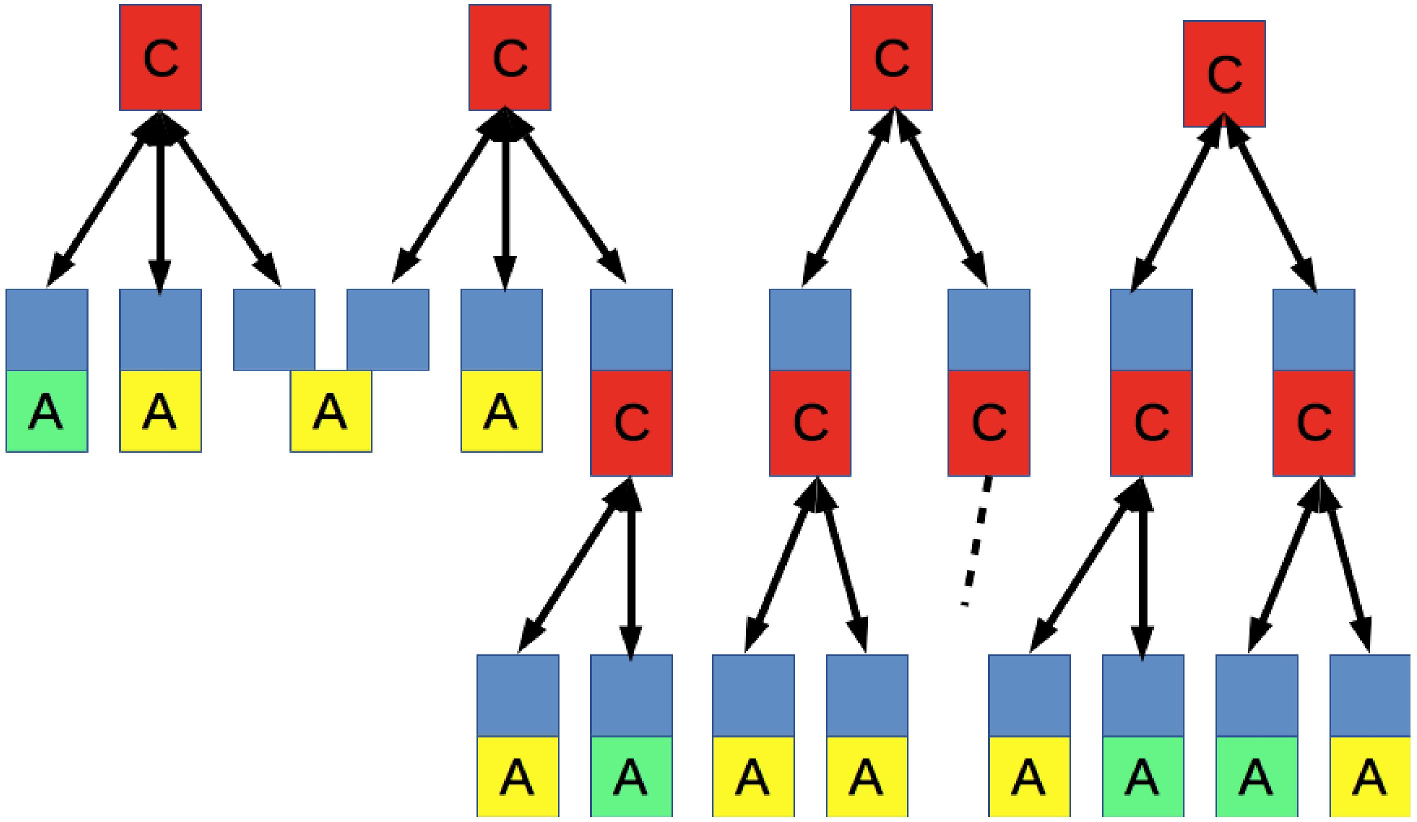
//Out-of-the-box Support for el. Contractors

- One of the main challenges is to support the electrical contractor best that has to show the devices and connections works BEFORE any commissioning happens
- We use a preprogrammed OGC group as „OOTB“ group that provides full tool-less contractor level checking of connections before any commissioning activity is in sight.
  - Luminaires will show connection success (by changing light output), and will show reaction to connected sensors
- Simple OOTB tools will provide more support in future, e.g. collecting and providing the count of the connected types and makes of equipment.
  - An additional challenge in IPv6 and especially RF networks is the basic network connectivity, that may need network commissioning to allow OOTB operation. This should be an attention point for future RF network developments.

//Use „Object- Stubs“ to enable diversity

- An actuator object must have the same ID throughout the group to enable group communication.
- But actuator details should change with future development and be different among competing vendors!
- We split the actuators into a logical object (a kind of stub), that provides a stable API for the OGC, and an executing object that may differ and that does not directly take part in group communication.
- This allows control objects to listen to „supergroup“ controls by means of a logical actuator object, and this way build the cornerstone of really seamless group hierarchies.
- Actuators that are part of more than one group use multiple logical objects, one for each group. This helps to easily implement priority schemes, and have a concise reporting also with „set, but not executed“ back to the group controls if needed.
- See more details in the public architecture document on www.openais.eu/en/results

//Logical Objects at work



//Wrap Up and Conclusion

- OpenAIS designed all architectural tools needed for professional lighting controls that at least match the available performance of best-in-class heritage lighting controls
- It is designed for a seamless integration into cloud based algorithmic control, data analysis and user control.
- In addition it allows for an open mix of vendors and third party services without loosing full interoperability.
- It provides user access to groups and their light points without the need to authenticate a user to each node.
- It enables fast response cloud based controls by means of „local agents“ (=stacked upon control objects) that do the „fast interaction“ locally.
- OpenAIS balances classical IoT integration and boosted local performance to a full win-win and seamlessly integrated environment.
- See www.openais.eu/en/results for the complete object model and many more details at the PUBLICLY accessible architecture documentation.

Please note we will have two focus groups after lunch in this room:

a) Focus on OGC performance, how it works and how it integrates. (Walter)
b) Focus on IoT integration, and why it provides unmatched benefits. (Ben)

Thank You!

www.openais.eu

OpenAIS

Open Architectures for Intelligent Solid State Lighting Systems

Supported by the Horizon 2020 funding of the European Union