Network Working Group   Kutscher
Internet-Draft   Ott
Expires: August 24, 2001   TZI, Universitaet Bremen
                                                        February 23, 2001

An Mbus Profile for Internet Appliance Control
draft-kutscher-mbus-ipac-00.txt

Status of this Memo

Copyright Notice

Abstract

This document discusses scenarios for the control of Internet Appliances -- Internet hosts with with specific user funtionalities -- using the Mbus protocol [1]. A first sketch of an Mbus application profile for controlling Internet appliances is presented, describing mechanisms for controlling a group of co-located appliances without the need for central controlling entities.

This document does not address the issue of wide area control, i.e., controlling appliances that are not on the same network link as the controlling entity. Instead, it is expected that the Mbus based local control is to be complemented by a, yet to be defined, protocol for that purpose.

The underlying message passing and addressing mechanisms for the

Mbus is defined in the Mbus transport specification[1].

This document is a contribution to the Internet Appliance Control (ipac) BoF at the 50th Internet Engineering Task Force meeting. Comments are solicited and should be addressed to the authors.

Table of Contents

1. Introduction

1.1 Background

The Mbus transport specification[1] defines the transport mechanisms of the Message Bus (Mbus), a local coordination infrastructure that allows message passing between a group of application components.

As its basic service, the Message Bus provides local (intra-system) exchange of messages between components that attach to the Mbus (Mbus entities). A system is typically expected to comprise one or more hosts, but a may also extend across a network link and include several hosts sharing the tasks; a local system does not extend beyond a single link -- the Mbus is not intended or designed for use as a wide-area control protocol. Wide-area control has significantly different requirements regarding trust-models and scalability and must deal with different problems regarding reliability and message transport delay.

The message transport service provides group- and point-to-point-communication. It does not offer all the features that are frequently found in protocols for coordinating distributed applications in general such as guaranteeing a global or causal ordering of delivered messages.

Given these deliberate restrictions in functionality it is important to note that the Mbus implies a component model where communication between components can be realized without services from a full-featured infrastructure for distributed applications.

Message exchange takes place using UDP: datagrams are sent either via unicast to a single entity or multicast to a locally scoped group. For unicast communications, message delivery is optionally performed reliably, with acknowledgements and retransmissions taking place at the Mbus transport layer. All group communication is performed unreliably. For deployment in scopes larger than link-local Mbus the multicast address (i.e., the multicast scope to use) can be configured accordingly, or bridging elements can be deployed that interconnect two or more Mbus domains.

For unicast communications, message delivery is optionally performed reliably, with acknowledgements and retransmissions taking place at the Mbus transport layer. Point-to-point and multicast communication is in general not distinguished by the transmission mechanism employed at the IP layer but rather by the qualification of the Mbus destination address. There is however the possibility to use IP-unicast for messages that are directed to a single receiver. (See section Entity Awareness.)

A key concept of the Mbus is its flexible and extensible addressing scheme. Mbus entities are identified by n-tuples with each component of the tuple represented as an attribute-value pair. The addressing scheme for conferencing applications includes address elements like conference (conf), media (media), module type (module), application name (app), and application identifier (id). For example:

(class:lamp location:bedroom floor:1 id:1035-0@192.168.1.1)

Each Mbus entity responds to messages addressed to any subset of its own address. For example, the entity with the address illustrated above will respond to messages pro-viding the following target addresses:

(class:lamp location:bedroom floor:1 id:1035-0@192.168.1.1)

(class:lamp location:bedroom id:1035-0@192.168.1.1)

(class:lamp id:1035-0@192.168.1.1)

(class:lamp)

()

A fully qualified address (with all components of the tuple present) indicates a unicast Mbus address. Messages with incomplete addresses have the potential to be received by multiple entities.

An entity on the Message Bus can learn the existence of other entities (and their complete addresses) by listening to the self-announcement messages that all entities send periodically. The rate at which these periodic heartbeat messages are sent is adapted dynamically depending on the number of entities in a Mbus session, thus allowing the Mbus to scale to larger groups of entities.

This mechanism is used to locate entities and to monitor their liveness during a session but also to build a complete set of the addresses of all available entities -- Mbus layer addresses and transport layer addresses. The latter information can be used for optimization strategies, e.g. for deciding whether a message can be sent via IP-unicast.

Based upon these awareness functions, a bootstrap procedure is defined that allows entities to determine whether all other entities they depend on are present (without bearing the risk of deadlocks).

Each entity has a unique Mbus address that can be used to identify it and that is composed of any number of named address elements. The types and values of address elements are application-specific and

can be used to aggregate entities into address groups.

Address element names may be associated with semantics: by providing certain address elements, entities can signal the type of service functionality they are able to supply. The Mbus specification itself does not impose any restrictions on application specific address elements. The semantics are provided by application specific profiles. For example, the address element set for conferencing applications defines elements for distinguishing entities by the media type that is assigned to them allowing a local conference controller to address one or more audio, video and other media tools.

A message that is to be sent via the Mbus has a target address -- at the application level -- that is used to determine how to deliver the respective message. This allows for subject-based addressing-like message delivery semantics and different communication models represented by the different group addressing features:

Broadcast: When a target address list is empty the message is broadcast to all entities on the Mbus.

Unicast: A message that contains a target address that is a unique Mbus address of another entity will be processed by this entity only. The Mbus defines mechanisms allowing such messages to be sent directly via unicast to the specific en-tity.

Multicast: As mentioned above target addresses can be used to identify groups on a per-message basis. All entities that match the given target address will receive and process corresponding messages: In situations where a certain module requires a specific service functionality, that can be pro-vided by more than one other module, it can use a multicast address specifying the group of service providers to locate the desired entity. This may facilitate the imple-mentation of service clients significantly: addresses of ser-vice providers do not need to be hardcoded and the com-munication model can accommodate many different spe-cific scenarios regardless of the number of potential service providers.

It is not necessary to know the exact addresses of all potential receivers of a message. Instead a sufficiently unam-biguous address list can be used. For example, in order to reach all audio engines in a session the address list (media:audio module:engine) might be appropriate.

Mbus messages are text-encoded and consist of a header with protool information and a payload section that can carry several textual

commands with parameter lists. Command names are hierarchical and a set of basic data types for command parameters is defined, including numeric types, character strings, lists and opaque data.

Message authentication and encryption are supported as inherent transport features to prevent malicious attacks and to provide privacy for the communication within an Mbus domain. This allows the Mbus to accommodate multiple sessions of a user per host (including cross-session coordination) as well as any number of users on the same host or link (preventing accidental cross-user interaction).

The Mbus guidelines[2] define a list of conventions for terminology, algorithms and procedures for higher level interaction models that are useful for applications using the Mbus. These conventions are intended as guidelines for designers of Mbus application profiles and Mbus implementations/applications.

This document builds on these two specifications and provides an Mbus application profile for Internet Appliance Control that uses the conventions codified in the Mbus guidelines[2] to specify an Mbus application profile, i.e., a list of Mbus commands and procedures that allow to implement Internet appliances and corresponding controllers.

1.2 Scope of this Document

This document discusses a first command set and corresponding interactions between application components for Internet appliance control, such as

- locating appliances;
- discovering capabilities and services of appliances;
- sending event notifications from appliances to controllers or other appliances; and
- sending controlling messages to appliances.

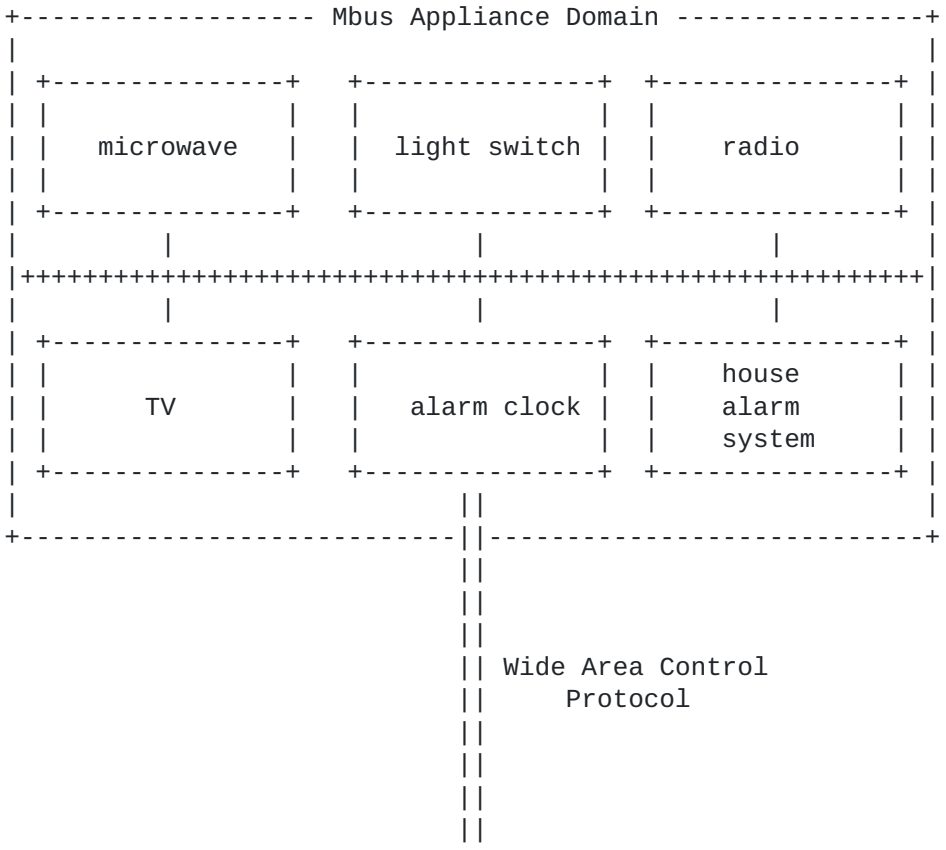
The Mbus commands presented are mainly intended as illustrations of the principle discussed here. Because the requirements of the application are not yet defined, we only present examples for the sake of clarification and for initiating discussions.

2. The System Model

The system model that is used in this document assumes that the appliances of a certain control domain, e.g., a house, build a cooperation group, that includes appliances (controllees) as well as controllers. However, a clear distinction between controllers and controllees is not always reasonable. For example, the event notifications generated by a certain appliance might be of interest to more than one entity, e.g., a controller. Instead, the corresponding information could be useful for a group of entities at the same time.

On the other hand, is is required to ensure that a certain appliance, say a microwave oven, is controlled by exactly one controller at a time, without the possibility of conflicts that could arise from receiving control messages from different entities.

We therefore distinguish between different types of interaction between appliances. Some interaction styles are appropriate for group communication, while others should be realized with the notion of tight, exclusive control relations.



As shown in the above picture, we assume that a group of appliances is modeled as an Mbus domain where a set of Mbus entities represent the appliances. It is further assumed, that wide area control is to be provided by a dedicated control protocol, e.g., a SIP-based protocol. See Section 5 for a disussion of the interworking with a wide area control protocol.

3. Characteristics

3.1 Addressing

The Mbus provides a subject-based addressing mechanism. This means, an appliance can use a descriptive address that contains information about service functionality it can provide. This information can be used for addresses of messages that are to be sent to the appliance. Using the wildcard/group communication mechanisms it thus is possible to address messages to an appliance by refering to a function (or other criteria).

There is another aspect of the Mbus addressing mechanism: Appliances that are generating periodic event notifications that would be sent to a well-known group target address. Other appliances can "tune in" into this event notification "channel" by choosing ("joining") a proper Mbus address. This is a scalable way to implement simple "subscribe/notify" scenarios, because the entities that generate the notifications do not have to know the exact list of subcribers.

3.2 Autoconfiguration

In general, it should be possible to connect a set of appliance systems without having to manually configure them. There might be some degree of configuration that is required to setup trust relationships etc. but apart from that, it is probably not acceptable to require customers to configure and maintain databases of IP addresses and other information about their appliances in order to be able to control and use them.

The Mbus can help to reduce the amount of manual configuration by its addressing architecture and the notion of a "soft state" of addressing information that is maintained at each Mbus entity:

As described in Section 3.1, entities are addressed using subject based addressing. This means, it is not always required to know the addresses of receivers exactly, because the group communication mechanisms allow senders to send messages to a group address.

Additionally, the entity awareness mechanism enables entities to build and maintain a list of active entities. Because of the periodic self-announcement messages that are sent by each entity, this state can be "soft", does not have to be initialized manually and is updated automatically.

3.3 Serverless Operation

In order to achieve plug-and-play functionality, robustness, device mobility and ease of deployment it is required to be able to have

appliance functioning correctly without having to supply a central server. A central server would have to be installed, maintained -- and may possibly fail at some time.

Due to the entity awareness mechanisms and the peer-to-peer communication model, the Mbus can accomodate serverless operation very easily: Mbus based appliances can be attached to a network and will immediately be able to communicate without the need of a central server. (It should be noted, though, that servers of any kind may be supplied to provide for (de-)centralized functionality).

3.4 Interaction Models

Within a group of appliances, more than the obvious client-server communication model will be required. The two most important communication styles:

Event notification: In an appliance environment, it is useful to have certain appliances asynchronously send certain information that other entities can take advantage of.
Using group communication and the notion of subject based addressing it is possible to realize this functionality.

RPCs: In order to control appliances reliably, it is required to invoke remote procedures and receive acknowledgements in order to obtain results and error notifications. For a message oriented communication infrastructure, this means that a mechanism for reliable message transmission and the possibility to relate reponses to message invocations is required. The Mbus therefore provides reliable message transmission of messages that are sent to a single entity. The [2] provide convention for building RPC communicatin upon this basic reliable message exchange.

4. Examples

In the following sections we present some scenarios and potential sample message exchanges that are primarily intended to illustrate the ideas behind appliance control via the Mbus.

4.1 Integrating Devices into an Appliance Network

Home Configuration:

1. Buy.

2. Install and "plug in" (provide local KEY for HMAC and possibly encryption).

3. Provide keys possibly use multi-level key provisioning s in Bluetooth or similar environments.

4. Determine device level parameters in browser.

5. Assign name and other home level parameters.
   This step could be automated if an individual "locator" components is available per logical location and there is no interference between these components. This could e.g. be realized by having an individual link in each room and a room-local multicast address.

Home level parameters can either be stored in the devices themselves, or, in the case of dumb devices, be stored externally. In that case, the devices could either be configured dynamically to use the home level address or proxy entities could be deployed that represent the entities using the home level addresses and relay messages to the device level addresses.

Functional components:

- Device manager (e.g. Mbus Browser)

- Device proxy

  * for non-Mbus entities

  * for very trivial entities

1. Tell the vendor when you buy it and have him configure it properly (plus remote maintenance service).

2. Use a small configuration tool (Mbus Browser) and store the settings in the device (in permanent storage).

3. Use a small configuration tool (Mbus Browser) and store the settings in some bootstrap device (for the device itself has only ephemeral storage) -> BOOTP/DHCP-style initialization protocol.

4. Provide some proxying mechanisms.

5. ...

Appearance on the Mbus:

```
mbus/1.0 U (id:4711@192.168.1.11 class:lamp) () ()
mbus.hello ()
ia.desrcription (vendor=Fasel name=lignerose-designer-lamp ...)
ia.caps (power)
ia.properties (power=100W bulbs=3 voltage=220V current=0.5A)
ia.status.power (off)
```

Query:

```
mbus/1.0 U (id:4711@192.168.1.11 class:lamp) () ()
mbus.whereami ()
```

Response:

```
mbus/1.0 U (class:configurator id:4711) (id:4711@192.168.1.11 class:lamp) ()
mbus.associate (id:com.manufacturer.lamp.23765827346593 floor:1 \
           location:bedroom name:my-reading-light)
```

Re-appearance on the Mbus:

```
mbus/1.0 U (id:4711@192.168.1.11 class:lamp floor:1 location:bedroom \
           name:my-reading-light) () ()
mbus.hello ()
```

4.2 Controlling Appliances

Controlling a lamp:

```
mbus/1.0 R (id:875462873694 class:remote-control) (id:4711@192.168.1.11
class:lamp) ()
power.on ()
power.dim (0.50)
```

Asynchronous status notification:

```
mbus/1.0 U (id:4711@192.168.1.11 class:lamp) (class:lamp status)
power.status (on dim=50)
```

Shutdown when leaving the house:

```
mbus/1.0 U (class:door name:front-door id:4711) (class:lamp)
power.off ()
```

Alternative: subcribe/notify

```
mbus/1.0 R (id:4711@192.168.1.23 class:monitor) (id:4711@192.168.1.11
class:lamp)
mbus.subscribe (power.status)

mbus/1.0 R (id:4711@192.168.1.11 class:lamp)  (id:4711@192.168.1.23
class:monitor)
power.status (on dim=50)
```

Independent provision and consumption of information:

```
mbus/1.0 U (class:environment id:76490947983 name:terrace-thermometer) \
           (class:environment) ()
weather.temperature ("25F")


mbus/1.0 U (class:environment id:76490947984 name:light-sensor)
(class:environment) ()
weather.light (brightness:0.25)


mbus/1.0 U (class:clock id:76490947985) (class:clock) ()
clock.current-time ("18:25:17 UTC")
clock.alarm-time (name="wakeup" time="07:00:00 UTC")
clock.alarm-time (name="call-mother-in-law" time="21:00:00 UTC")


mbus/1.0 U (class:remote-control id:4774856923649) (class:clock id:
76490947985) ()
clock.alarm.set (name="coffee" time="06:50:00 UTC" address="class:coffee-
maker" \
             command="coffee.brew (\"type=strong\")")


mbus/1.0 U (class:clock id:76490947985) (class:clock) ()
clock.current-time ("18:25:17 UTC")
clock.alarm-time (name="wakeup" time="07:00:00 UTC")
clock.alarm-time (name="call-mother-in-law" time="21:00:00 UTC")
clock.alarm-time (name="coffee" time="06:50:00 UTC")


mbus/1.0 U (class:remote-control id:4774856923649) (class:clock id:
76490947985) ()
clock.alarm.delete (name="call-mother-in-law")
```

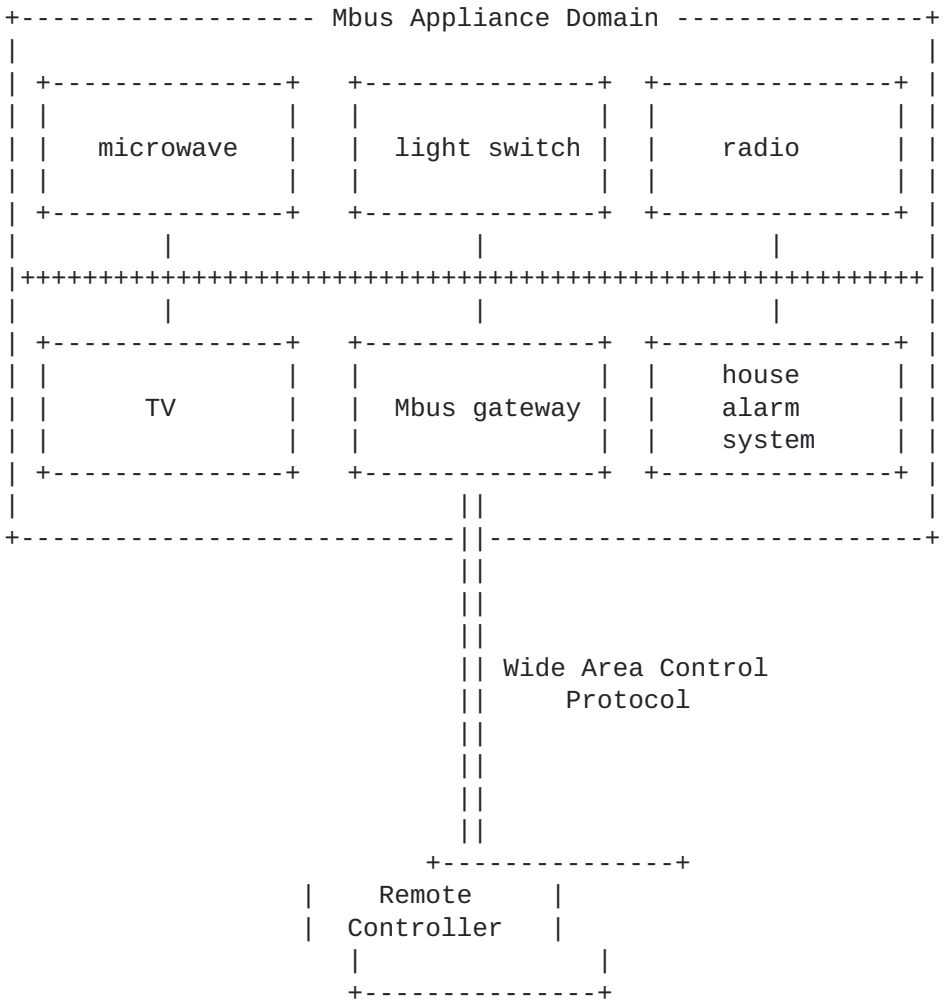
Alarm at 06:50:

```
mbus/1.0 U (class:clock id:76490947985) (class:coffee-maker) ()
coffee.brew ("type=strong")
```

5. Interworking with Wide Area Control

Since the Mbus-based control of appliances is defined for local network of appliances only the issue of how to provide remote control from arbitrary Internet hosts need to be addressed.

We propose the use of gateway systems that translates requests from a controller to Mbus messages. The protocol that is used for wide area control is not defined in this document. It could be HTTP or a SIP variant. A few required properties for such an protocol can be given.



6. Integrating Dumb Devices

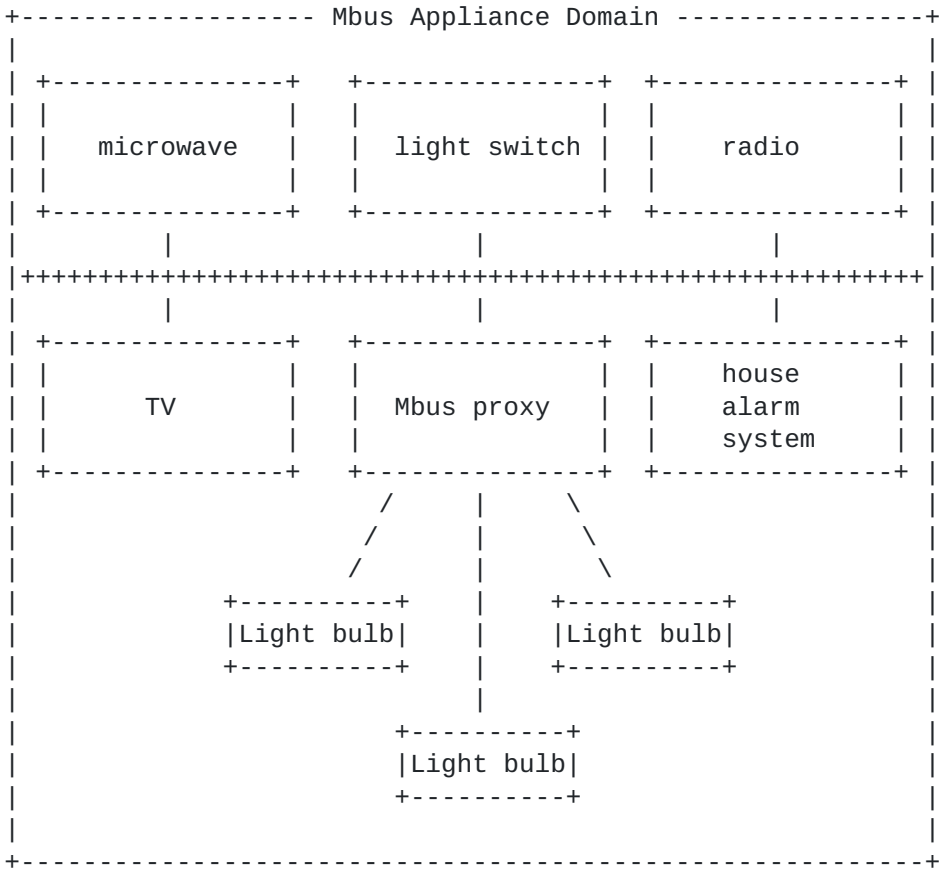
Not every device that needs to be controlled in an appliance network can be equipped with a TCP/IP and an Mbus stack. There may be cost issues for very small devices or the issue of legacy devices that users want to be able to integrate.

Since a host with an Mbus stack can host more than one Mbus entity that are uniquely addressable, the proposed solution for these scenarios is to employ "proxy" Mbus systems that simply represent the dumb devices that cannot directly connect to the Mbus.

In fact, the presence of such a proxy system is completely transparent to the other entities of an Mbus session since it is not a special to have multiple entities reside on one host.

The way how those dumb devices are connected to and controlled by an proxy system is of course left to the implementations and is not subject of this document.

The following figure illustrates this using the example of a proxy system for a set of light bulbs.


Mbus Appliance Domain
microwave
light switch
radio
TV
Mbus proxy
house
alarm
system
Light bulb
Light bulb
Light bulb

References

[1] Ott, J., Perkins, C. and D. Kutscher, "A Message Bus for Local Coordination", Internet Draft draft-ietf-mmusic-mbus-03.txt, December 2000.

[2] Kutscher, D., "The Message Bus: Guidelines for Application Profile Writers", Internet Draft draft-ietf-mmusic-mbus-guidelines-00.txt, February 2001.

[3] Handley, , Schulzrinne, H., Schooler, and Rosenberg, "SIP: Session Initiation Protocol", Internet Draft draft-ietf-sip-rfc2543bis-02.txt, November 2000.

Authors' Addresses

Dirk Kutscher
TZI, Universitaet Bremen
Bibliothekstr. 1
Bremen 28359
Germany

Phone: +49.421.218-7595
Fax: +49.421.218-7000
EMail: dku@tzi.org

Joerg Ott
TZI, Universitaet Bremen
Bibliothekstr. 1
Bremen 28359
Germany

Phone: +49.421.201-7028
Fax: +49.421.218-7000
EMail: jo@tzi.org

Full Copyright Statement

Acknowledgement

Funding for the RFC editor function is currently provided by the Internet Society.