iotops
Internet-Draft
Intended status: Informational
Expires: 9 May 2026

J. Romann, Ed.
Universit辰t Bremen
H. Damer, Ed.

J. Jim辿nez
Ericsson

5 November 2025

Using MUD in Constrained Environments
draft-romann-iotops-mud-constrained-00

Abstract

This document specifies additional ways for discovering and emitting Manufacturer Usage Descriptions (MUD), especially in constrained environments, utilizing the Constrained Application Protocol (CoAP), CoRE Resource Discovery, and CBOR Web Tokens.

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-romann-iotops-mud-constrained/.

Source for this draft and an issue tracker can be found at https://github.com/namib-project/draft-coap-mud.

Status of This Memo

Table of Contents

1. Introduction and Overview

Manufacturer Usage Descriptions (MUDs), which have been specified in [RFC8520], provide a means for end devices to indicate to the network what sort of access and network functionality they require to properly function. This information enables automatic configuration of firewall appliances to limit device network access to the specified network resources only, which in turn reduces the attack surface for MUD-capable devices.

While [RFC8520] contemplates the use of CoAP-related [RFC7252] policies, the MUD URL discovery methods it specifies (DHCP/DHCPv6, LLDP, and X.509 certificates) are not well-suited for constrained environments (e.g., 802.15.4 networks using 6LoWPAN and SLAAC).

Therefore, this document introduces a number of additional ways for distributing MUD URLs using CoAP -- such as well-known URIs and parameters for the CoRE Link-Format -- which are better suited for constrained devices. Furthermore, this document specifies a method of encoding MUD URLs in (signed) CBOR Web Tokens (CWTs) [RFC8392], which allows MUD managers to better validate the authenticity of both the URL itself and the associated MUD file.

1.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This specification re-uses the terminology defined in [RFC8520].

2. Architecture

Building upon the MUD architecture specified in [RFC8520], there are two main network components relevant for this document: The _Thing_ that wants to obtain network access via a Router or Switch, and the _MUD Manager_ that processes MUD URLs, retrieves MUD files from the MUD file server, and configures other network components accordingly. For the purposes of this document, we extend the MUD architecture with another component: The _MUD Receiver_. The MUD Receiver is the device responsible for providing and/or performing CoAP requests to

resources intended for MUD URL delivery. While this component can also be a part of the same physical device as the MUD Manager or the router/switch, this is not required. Both the Thing and the MUD Receiver can play active roles when using CoAP [RFC7252] for exposing and discovering MUD URLs.

A general overview of the MUD architecture adjusted for using CoAP in a constrained environment can be seen in Figure 1. Here, we can see that both the Thing and the MUD Receiver (as the recipient of the MUD URLs) may initiate the MUD discovery process: The Thing can contact and register with MUD URL recipients, e.g. by sending a CoAP POST request via Multicast and/or addressing a well-known registration endpoint. Conversely, a MUD Receiver can initiate the discovery process, e.g. by sending a CoAP GET request to a well-known URI via multicast.

Note that the protocol used for communication between the MUD Receiver and MUD Manager is considered out of scope for this document and is considered implementation-specific.

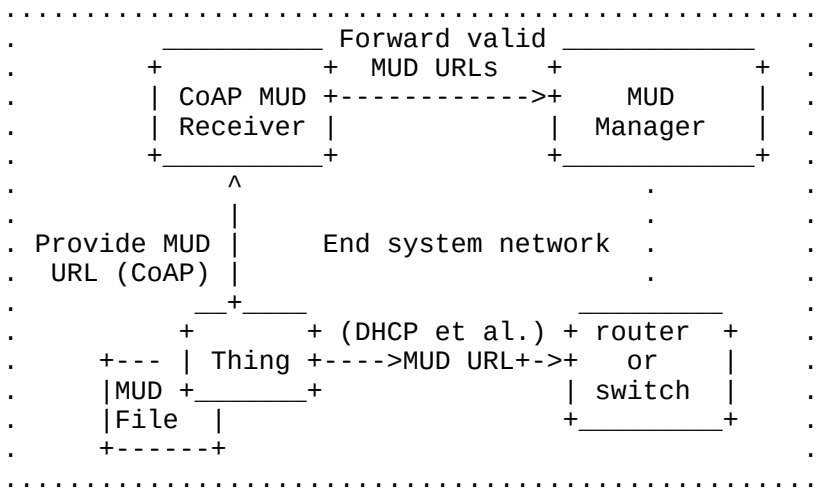

Figure 1: Exposing and discovering MUD URLs via CoAP

Optionally, Things may also provide additional means for proving the authenticity of the MUD URL associated with them. For this purpose, this document specifies how to use CBOR Web Tokens [RFC8392] to include MUD URLs as part of a signed and therefore authenticable token.

2.1. Methods of MUD URL Distribution

In general, we consider two mechanisms for exposing MUD URLs in a constrained network: Using dedicated resources and embedding into existing self description formats.

Things can expose MUD URLs as a dedicated resource, which may ease discovery of MUD-capable Things through said resource and enables the transmission of arbitrary payload types, including ones that provide authentication (e.g., CWTs [RFC8392]).

Alternatively, MUD URLs can also be referenced through other self description formats such as SUIT manifests [I-D.ietf-suit-mud]. Including MUD URLs with other self descriptions can be advantageous with regard to the availability of said information (e.g., by making the MUD URL available through a central directory). If the other self description format or its method of distribution provides means for authentication, they may also be used for validating MUD URLs.

In this specification, we will describe how to embed a MUD URL into a CoRE Link Format [RFC6690] resource, which may itself be retrieved directly from the device or through a CoRE Resource Directory [RFC9176]. Additionally, we will define dedicated CoAP resources both for providing and receiving MUD URLs.

2.2. MUD URL CoAP Submission Flows

In general, this specification provides two ways by which a MUD URL transmission can be performed using a dedicated CoAP resource.

In environments where many Things need to be managed over several subnets and where multicast usage is not desirable, it may be advantageous if MUD URLs are submitted by the Thing through a CoAP resource provided by the MUD Receiver. This will be referred to as the "Thing-initiated" submission flow for the remainder of this specification.

Conversely, in environments where multicast is not an issue and things might be limited in their capabilities, it may be easier for the MUD Receiver to retrieve the MUD URL from a CoAP resource provided by the Thing. In this specification, this will be referred to as the "Receiver-initiated" submission flow.

2.2.1. Using the MUD URL Resource (Receiver-initiated)

In the Receiver-initiated flow, Things provide a CoAP resource discoverable by the means provided in Section 2.5, which is then requested by MUD Receivers to retrieve the MUD URL.

In general, the Receiver-initiated MUD URL flow can be divided into these steps:

1. After joining the network, the Thing starts providing a CoAP resource to retrieve the MUD URL. This resource should provide the MUD URL in one of the formats specified in Section 2.3. It also makes this resource discoverable for MUD Receivers using the methods specified in Section 2.5.

2. The MUD Receiver discovers the resource using the aforementioned methods, e.g., by periodically requesting a well-known URI using multicast.

3. The MUD Receiver retrieves the discovered resource for devices for which the MUD Controller does not have a current MUD URL. To do so, it performs a CoAP request for the discovered MUD URL resource URI using the GET method, which is responded to with the appropriate payload. Receivers MUST specify their desired payload format using the Accept option Section 5.10.4 of [RFC7252]. During content negotiation, Receivers MUST start with requesting the Content-Format that provides the greatest degree of authenticity protection (i.e., prefer CWTs over plaintext transmission).

2.2.2. Using the MUD URL Submission Resource (Thing-initiated)

In the Thing-initiated flow, Things discovery a submission resource provided by the MUD Receiver and submit their MUD URLs to this resource.

This flow can be divided into these general steps:

1. The MUD Receiver provides a CoAP resource that Things can submit their MUD URLs to. It also makes itself discoverable for Things using the methods specified in Section 2.5.

2. The Thing connects to the network. After connecting, it discovers the MUD URL submission resource using the aforementioned methods.

3. The Thing submits the MUD URL to the previously discovered URI. To do so, it performs a CoAP request to the discovered URI with the POST method. The MUD URL is contained as the message payload in this request using one of the content formats defined in Section 2.3. Receivers MAY be configured by the network administrator to limit the accepted Content-Formats to ones that provide some level of authenticity for the MUD URL. However, receiver implementations MUST also support unauthenticated MUD URL transmissions unless said policy forbids it.

2.3. MUD CoAP Payloads

For the purposes of this specification, we will define two formats for transmitting MUD URLs, which are suitable for different environments. MUD Receivers that conform to this specification MUST support both formats.

2.3.1. Plain URL

The easiest method of transmitting MUD URLs is using a plain text payload containing only the MUD URL. While this method has the advantage of simplicity, it does not contain any additional information that could be used by a MUD Receiver to authenticate the supplied MUD URL.

CoAP requests and responses that use this format MUST use the Content-Format option with the value corresponding to the "application/mud-url+plain" media type.

2.3.2. MUD URLs inside of CBOR Web Tokens

Previous methods of transmitting MUD URLs do not allow for authentication of supplied MUD URLs. To accomodate for environments where authentication of MUD URLs is desired, it is also possible to include the MUD URL as a claim inside of a CBOR Web Token [RFC8392]. This allows for MUD Receivers or MUD Controllers to verify the authenticity of the provided MUD URL, given that the key used for signing the CWT is known to belong to the Thing’s manufacturer.

CBOR Web Tokens that contain MUD URL information have the following properties:

* The MUD URL is contained as an ASCII-encoded string in the mud-url claim.

* The Token MAY contain proof-of-possession claims [RFC8747]. If it does, the MUD Receiver MUST verify that the Thing is in possession of the key specified in the cnf claim (see Section 2.4).

* The Token MAY contain an expiry time. If an expiry time is specified, the MUD URL should be resubmitted or requested again shortly before the original CWT expires. Note that using an expiry time could cause problems if the device is unable to perform a refresh, e.g., due to a power outage.

CoAP requests and responses that use this format MUST use the Content-Format option with the value corresponding to the application/mud-url+cwt media type.

2.4. Proof-of-Possession for CBOR Web Tokens with MUD URLs

If the MUD URL is transmitted as a CBOR Web Token that includes proof-of-possession claims, that claim MUST be verified. In general, most already specified PoP methods aim to reduce overhead by combining proof-of-possession with the authenticity-, integrity-, and replay-protection of the underlying CoAP session. Some examples of this approach may be found in the specifications for ACE-OAuth profiles found in [RFC9202], [RFC9203], and [RFC9431].

This specification provides two different methods for performing proof-of-possession in Section 3.3, which are adapted from the ACE-OAuth DTLS profile [RFC9202] and OSCORE profile [RFC9203], respectively. Additionally, we will describe the necessary steps required for defining additional proof-of-possession methods in TODO.

2.5. Resource Discovery

In this section, additional methods for resource discovery in constrained environments are defined.

2.5.1. Well-known URI and Multicast Addresses

This document introduces a new well-known URI for discovering MUD URLs directly: /.well-known/mud-url.

/.well-known/mud-url MAY be used to expose a URL pointing to a MUD file hosted by an external MUD file server. This MUD file MUST describe the device the URL was retrieved from or is referring to within a list of CoRE links.

[RFC7252] registers one IPv4 and one IPv6 address each for the purpose of CoAP multicast. In addition to these already existing "All CoAP Nodes" multicast addresses, this document defines an additional "All MUD CoAP Nodes" IPv6 multicast addresses that can be used to address only the subset of CoAP Nodes that support MUD. If a device exposes a MUD URL via CoAP, it SHOULD join the respective multicast groups for the IP versions it supports. Lastly, this

document also defines an additional "All MUD Receivers" IPv6 multicast address that can be used by Things to interact with MUD Receivers in their vicinity.

2.5.2. CoRE Link Format and CoRE Resource Directories

Resources which either host MUD URLs or MUD files MAY be indicated using the CoRE Link Format [RFC6690]. For this purpose, additional link parameters are defined: With the resource-types mud-file and mud-url, a link MAY be annotated as pointing to a MUD file or a MUD URL, respectively. If a MUD URL is included as a resource in a list of CoRE web links, the supported Content-Formats MUST be indicated using the ct parameter.

MUD Receivers or other devices can send a GET requests to a CoAP server for /.well-known/core and get in return a list of hypermedia links to other resources hosted in that server, encoded using the CoRE Link-Format [RFC6690]. Among those, it will get the path to the resource exposing the MUD URL, for example /.well-known/mud-url and resource-types like rt=mud-url.

Things SHOULD only provide at most one link of resource type mud-url and mud-file each, unless they use MUD URLs encoded as CWTs with expiry times and provide an additional, basic MUD URL to enable token refresh. However, if a CoRE Link Format description contains multiple mud-url or mud-file entries of the same type, MUD Receivers MUST choose the MUD URL that provides the highest degree of authenticity protection (where CWTs with expiry time take precedence over CWTs without expiry time). If multiple MUD URLs with the same level of authenticity protection exist, the MUD Receiver SHOULD try them in an arbitrary order until one is successfully validated.

By using CoRE Resource Directories [RFC9176], devices can register a MUD file or MUD URL and use the directory as a MUD repository, making it discoverable with the usual RD Lookup steps. A MUD manager itself MAY also act as a Resource Directory, directly applying the policies from registered MUD files to the network. In addition to the registration endpoint defined in [RFC9176], MUD managers MAY define a separate registration interface when acting as a CoRE RD.

3. Obtaining a MUD URL via dedicated CoAP Resources

With the additional mechanisms for finding MUD URLs introduced in this document, MUD managers can be configured to play a more active role in discovering MUD-enabled devices. Furthermore, IoT devices could identify their peers based on a MUD URL associated with these devices or perform a configuration process based on the linked MUD file’s contents. However, the IoT devices themselves also have more

options for exposing their MUD URLs more actively, using, for instance, a MUD manager's registration interface.

In the remainder of this section, we will outline potential use-cases and procedures for obtaining a MUD URL with the additional mechanisms defined above.

3.1. Thing Behavior

This section outlines specifics of this specification regarding the behavior of Things.

3.1.1. Exposing MUD URLs

Things MAY expose their MUD URL using a dedicated resource hosted under /.well-known/mud-url. If a MUD URL is exposed this way, the resource MUST offer at least one of the specified serialization methods and MUST allow clients to choose between them using the Accept option, if provided. If the Thing supports resource discovery using a /.well-known/core resource, MUD URL resources SHOULD be advertised there using the CoRE Link-Format [RFC6690], indicating the available Content-Formats using the ct parameter.

3.1.2. Registering MUD URLs with MUD Receivers

Things MAY actively register their MUD URLs with MUD Receivers offering a registration interface. To perform the registration process, Things can perform a discovery process using the CoRE Link Format or directly include their MUD URL as a payload of a CoAP POST request. If Things are actively registering their MUD URLs with MUD managers, they SHOULD regularly repeat the process to keep interested parties informed about their presence and their associated MUD URL.

Using the CoRE Link Format, Things MAY send a GET request to the All CoAP Nodes (IPv4) or the All MUD Receivers (IPv6) multicast address, requesting the /.well-known/core resources and including an (optional) query parameter rt=mud.url-register. After filtering the obtained links for the Resource Type mud.url-register to identify available submission interfaces, the Thing MAY then send a unicast POST request to the discovered MUD manager, containing the MUD URL as a payload and a corresponding Content-Format option. Alternatively, a Thing MAY also use a CoRE Resource Directory [RFC9176] to perform the discovery process.

Things MAY also directly send a POST request containing the MUD URL as a payload and a corresponding Content-Format option via unicast or to the All CoAP Nodes (IPv4) or the All MUD Receivers (IPv6) multicast address, using the well-known URI /.well-known/mud-submission.

3.1.3. Finding MUD URLs of Other Things

If a Thing should be interested in the MUD URLs of one or more of its peers, it MAY use the same mechanisms specified for MUD Receivers described in the following section.

3.2. Receiver Behavior

MUD Receivers are assumed to be mostly non-constrained devices. Accordingly, this specification puts most of the implementation burden regarding support for flows and formats on the receivers, while keeping the requirements for Things as small as possible. In general, it is recommended that MUD Receivers support as much of the specification as possible in order to support as many different Things as possible.

3.2.1. Discovery

For the discovery process described in Section 2.5, the following considerations apply to MUD Receivers:

* MUD Receivers that support IPv6 SHOULD regularly perform a CoAP request to the "All MUD CoAP Nodes" multicast address for the /.well-known/mud-url URI.

* MUD Receivers that support IPv4 SHOULD regularly perform a CoAP request to the "All CoAP Nodes" multicast address for the /.well-known/mud-url URI.

* MUD Receivers that support IPv6 devices MUST join the "All MUD Receivers" IPv6 multicast group.

* MUD Receivers that support IPv4 devices MUST join the "All CoAP Nodes" IPv4 multicast group.

* MUD Receivers SHOULD regularly query any CoRE Resource Directories relevant for the subnet they are responsible for.

* MUD Receivers SHOULD register their submission resource to any CoRE Resource Directories relevant for the subnet they are responsible for.

3.2.2. MUD URL Submission Resource

This section describes the behavior for MUD Receivers regarding the Thing-initiated submission flow:

* MUD Receivers MUST provide a submission resource under the /.well-known/mud-submission well-known URI.

* MUD Receivers SHOULD include their submission resource in any CoRE Link Format descriptions of their resources (both in /.well-known/core and in CoRE RD registrations, if applicable).

* MUD Receivers MAY indicate failure of MUD URL submission using a CoAP Error Code. If the submitted MUD URL is encoded in a CBOR Web Token including proof-of-possession claims, MUD Receivers MUST return the appropriate error code to initiate the proof of possession flow as described in Section 2.4.

3.2.3. MUD URL Resource

This section describes considerations for MUD Receivers regarding the Receiver-initiated submission flow:

* MUD Receivers MUST request MUD URLs from any resources known to them.

* MUD Receivers MUST re-request MUD URLs submitted as a CWT claim if the CWT has an expiry time that passed.

3.2.4. MUD URL Payload

Regarding the actual MUD URL payload transmitted using CoAP, the following considerations apply:

* MUD Receivers SHOULD treat devices for which MUD URL retrieval failed as devices the same way as devices that do not provide a MUD URL at all, unless an explicit differing policy is defined.

* MUD Receiver implementations MUST support the plain MUD URL payload, although support for these MAY be disabled by policy.

* MUD Receivers MUST support the CWT MUD URL claim.

* MUD Receivers SHOULD be configured with a policy as to which signers are authorized to sign tokens.

* MUD Receivers SHOULD support proof-of-possession semantics in CBOR Web Tokens and validate them before treating submitted MUD URLs using this format as valid.

3.3. Proof-of-Possession Methods

TODO

3.3.1. Requirements for Proof-of-Possession

TODO

3.3.2. Proof-of-Possession using DTLS

TODO

3.3.3. Proof-of-Possession using OSCORE

TODO

4. Security Considerations

(Very WIP for now)

The security considerations from [RFC8520] also apply to this document. Regarding CoAP specifics, [I-D.irtf-t2trg-amplification-attacks] provides information on possible attack scenarios. Additionally, the following considerations should be taken into account.

For Thing manufacturers that intend to implement this specification:

* It is recommended to use CBOR Web Token-encoded MUD URLs whereever possible to allow for better integrity checking

* Using expiry times for CWTs containing MUD URLs can be advantageous, but also has its shortcomings. Most notably, expiry times are not recommended if it is possible for Things to reach a state where they have neither the means to update their CWT nor a non-expired token in their persistent storage As devices may be out of service for longer periods of time, preventing them from refreshing CWTs, this state is unavoidable for the vast majority of devices.

* In order to use expiry times anyways, manufacturers could use two different CWTs: A backup CWT without an expiry time with a MUD URL pointing to a very restricted MUD file that only allows updating the main CWT, and a main CWT that has an expiry time but has a more broad MUD file referenced in it.

For network operators: Network operators SHOULD specify a policy that describes:

* Whether to accept unauthenticated MUD URLs (those that were not submitted as part of CWTs).

* The keys whose signatures should be accepted by the MUD Receiver if a submission using CWTs is performed.

5. IANA Considerations

* CoAP Resource Media Types Registry

* CoAP Content Format Registry

* well-known URI Registry

* IPv6 Multicast Address Registry

5.1. Well-Known ’mud-url’ URI

5.2. Well-Known ’mud-file’ URI

5.3. Well-Known ’mud-submission’ URI

5.4. New ’mud’ Resource Type

5.5. Media Types Registry

* application/mud-url+plain

* application/mud-url+cwt

5.6. CoAP Content-Format Registry

* application/mud-url+plain

* application/mud-url+cwt

6. Normative References

[I-D.ietf-suit-mud]
Moran, B. and H. Tschofenig, "Strong Assertions of IoT Network Access Requirements", Work in Progress, Internet-Draft, draft-ietf-suit-mud-10, 3 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-suit-mud-10>.

[I-D.irtf-t2trg-amplification-attacks]
Mattsson, J. P., Selander, G., and C. Ams端ss, "Amplification Attacks Using the Constrained Application Protocol (CoAP)", Work in Progress, Internet-Draft, draft-irtf-t2trg-amplification-attacks-05, 18 June 2025, <https://datatracker.ietf.org/doc/html/draft-irtf-t2trg-amplification-attacks-05>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/rfc/rfc2119>.

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, DOI 10.17487/RFC6690, August 2012, <https://www.rfc-editor.org/rfc/rfc6690>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/rfc/rfc7252>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/rfc/rfc8174>.

[RFC8392] Jones, M., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "CBOR Web Token (CWT)", RFC 8392, DOI 10.17487/RFC8392, May 2018, <https://www.rfc-editor.org/rfc/rfc8392>.

[RFC8520] Lear, E., Droms, R., and D. Romascanu, "Manufacturer Usage Description Specification", RFC 8520, DOI 10.17487/RFC8520, March 2019, <https://www.rfc-editor.org/rfc/rfc8520>.

[RFC8747] Jones, M., Seitz, L., Selander, G., Erdtman, S., and H. Tschofenig, "Proof-of-Possession Key Semantics for CBOR Web Tokens (CWTs)", RFC 8747, DOI 10.17487/RFC8747, March 2020, <https://www.rfc-editor.org/rfc/rfc8747>.

[RFC9176] Ams端ss, C., Ed., Shelby, Z., Koster, M., Bormann, C., and P. van der Stok, "Constrained RESTful Environments (CoRE) Resource Directory", RFC 9176, DOI 10.17487/RFC9176, April 2022, <https://www.rfc-editor.org/rfc/rfc9176>.

[RFC9202] Gerdes, S., Bergmann, O., Bormann, C., Selander, G., and L. Seitz, "Datagram Transport Layer Security (DTLS) Profile for Authentication and Authorization for Constrained Environments (ACE)", RFC 9202, DOI 10.17487/RFC9202, August 2022, <https://www.rfc-editor.org/rfc/rfc9202>.

[RFC9203] Palombini, F., Seitz, L., Selander, G., and M. Gunnarsson, "The Object Security for Constrained RESTful Environments (OSCORE) Profile of the Authentication and Authorization for Constrained Environments (ACE) Framework", RFC 9203, DOI 10.17487/RFC9203, August 2022, <https://www.rfc-editor.org/rfc/rfc9203>.

[RFC9431] Sengul, C. and A. Kirby, "Message Queuing Telemetry Transport (MQTT) and Transport Layer Security (TLS) Profile of Authentication and Authorization for Constrained Environments (ACE) Framework", RFC 9431, DOI 10.17487/RFC9431, July 2023, <https://www.rfc-editor.org/rfc/rfc9431>.

Authors' Addresses

Jan Romann (editor)
Universit辰t Bremen
Germany
Email: jan.romann@uni-bremen.de

Hugo Hakim Damer (editor)
Germany
Email: hdamer@uni-bremen.de

Jaime Jim辿nez
Ericsson
Phone: +358-442-992-827
Email: jaime@iki.fi