

Project Number: **4007 ( RE)**

Project Title: **Multimedia Education and Conferencing Collaboration over ATM Networks and Others (MECCANO)**

Deliverable Type: (PU/LI/RP)\* **PU**

Deliverable Number: **D6.1**

Contractual Date of Delivery: **31 March 1999**

Title of Deliverable: **The gateways and relays in MECCANO Release 1.**

Work-Packages contributing to the Deliverable: **6**

Nature of the Deliverable: (PR/RE/SP/TO/OT)\*\* **TO/RE**

Authors: **Joerg Ott, Universitaet Bremen, Colin Perkins, UCL**

### **Abstract:**

We describe the gateway architecture being developed in this workpackage as an extension of the Mbus work ongoing in WP7, and the initial demonstrator applications that leverage and build on the media engines developed in WP4. Two demonstrators have been delivered already: the UCL Transcoding Gateway (in D4.1) and AudioGate (in D4.2). We note directions for future work in the area of demonstrators: StarGate and enhanced versions of AudioGate and the UCL Transcoding Gateway.

### **Keyword list:**

demonstrators, gateway, Mbone, multicast, multimedia conferencing, relay, videoconferencing

---

\* Type: PU - public, LI - limited, RP - restricted

\*\* Nature: PR - Prototype, RE - Report, SP - Specification, TO - Tool, OT - Other

# 1. Introduction

Workpackage 6 of the MECCANO project deals with the development of various types of gateways:

- AudioGate, the Mbone-to-Telephony gateway provides audio access to unicast or multicast sessions on the Mbone for users connected from the PSTN, ISDN, or GSM.
- StarGate, the Call Signaling and Conference Control gateway is intended be able to translate arbitrary call/conference control protocols into one another – with SIP, H.323, ISDN as the primary focus for call signaling conversion.
- The UCL Transcoding Gateway (UTG) performs multicast-to-unicast and IPv6-to-IPv4 conversion, and optional transcoding between multiple encoding formats.
- A multicast to unicast relay that may be remotely controlled / configured through RTSP.

All these gateways vary in the range of functionality they provide and their respective complexity but they are all based on the same architectural concept. The Robust-Audio Tool (RAT) developed in the context of WP4 acts as the core media handling engine, but others may be developed or adapted to fit the Mbus architecture. Different Mbus control modules (which are largely developed in the context of WP7) are combined to control the gateway actions.

The different gateways and gateway components are currently at various stages of development. This deliverable gives an overview of a common gateway architecture – which is embedded in the overall MECCANO system architecture – irrespective of the different components’ state and then addresses the implementation of gateways and relays in more detail. It is a companion to MECCANO deliverable 7.1, and is expected to be read in conjunction with that deliverable.

Section 2 outlines the gateway, relay, and transcoder architecture and describes the roles of the various components and the Mbus as supporting infrastructure. Section 3 describes version 0.1 of AudioGate that was part of the software deliverable D4.2, section 4 provides an outline of the concepts for StarGate (which has not been delivered yet). Section 5 describes the current state of implementation of the UCL Transcoding Gateway – which was also included in D4.2. Finally, section 6 concludes this deliverable with a brief assessment of current results and a summary of potential (current and future) areas of application for the gateways and relays.

# 2. Gateway Architecture

The gateway architecture follows the overall system architecture to be outlined in D3.1 and employs the components and control mechanisms described in D7.1.

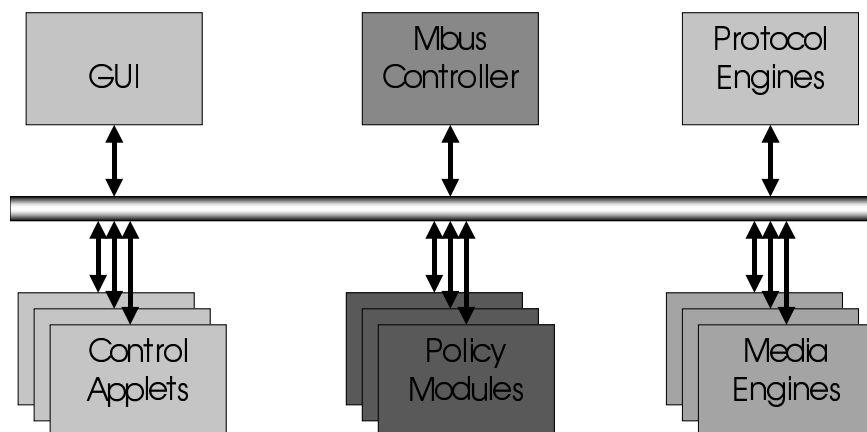


Figure 1: Overall Mbus Architecture for Endpoint, Gateways, and Management Systems

An overview of this system architecture is depicted in figure 1 with the following Mbus entities being envisioned so far:

- Media engines providing the functionality necessary for telecooperation (such as audio or video communications, shared workspace or editor, etc.);
- Control (protocol) engines managing interactions with remote users (e.g. providing means for call/conference setup, floor control, mutual awareness in a conference, etc.);
- graphical user interfaces (GUIs) as suitable means for a human user to access the system functionality implemented by the (control and media) engines;
- Policy modules that control automated system behavior (e.g. based upon user preferences, administrative settings, call/conference processing scripts, etc.);
- Applets that provide an abstraction from specific implementations of background/backend services (such as address resolution, certificate validation, user authentication, directory services, etc.); and
- An Mbus controller that may combine any number of the aforementioned modules and add specific interpretation/processing to create a particular integrated system type.

Gateways are designed based upon the Mbus concept. They consist of a set of components attaching to the Mbus with one dedicated Mbus controller defining the type of gateway and the others fulfilling well-defined pieces of the overall functionality the gateway provides. The building block approach and the types of components involved in the MECCANO gateways are described in the following subsection. Subsequently, the role of the Mbus as a transparent mechanism to control gateways is outlined as are the control Mbus components, followed by considerations on handling of media streams in gateways.

The Mbus infrastructure is described in the companion deliverable, D7.1, and is not described in detail here. For the purposes of this deliverable, it should suffice to note that the Mbus provides a uniform coordination channel over which applications may communicate to form a single logical endpoint in either an end-system or a gateway. For more details, the reader is referred to deliverable 7.1 which includes a full description of the Mbus along with complete protocol specifications.

The following sections then present the three types of gateways envisioned and partly implemented today, AudioGate, StarGate, and the UCL Transcoding Gateway.

## **2.1 Building Block Approach**

As with all other complex systems developed in the MECCANO project, gateways combine a set of Mbus entities to provide the desired system functionality. Out of the aforementioned categories, the currently developed gateways:

- use media engines for transcoding purposes but not for local capturing / replay of media streams;
- use one or more control protocol engines to interpret and encode messages of the respective protocols;
- optionally may make use of graphical user interface components to provide status information to an administrator and allow the administrator to configure the gateway, intervene with ongoing calls if necessary, etc. (however, none of the gateways currently do so);
- do not make use of policy modules (at this point in time);
- employ control applets for modular extensions to the core functions of the gateway; and
- implement an Mbus controller that actually implements the signaling conversion, invokes transcoding as necessary, etc.

The following subsections briefly outline the status of the generic Mbus entities implemented so far while the sections three to five then describe the respective gateways' Mbus controller and how the various generic Mbus entities fit together.

## **2.2 Mbus Entities for control**

Within MECCANO, the following control protocol entities have been developed so far:

- SIP engine – a fully Mbus capable SIP engine (based upon RFC 2543 but also starting to include the latest extensions currently only specified as Internet Drafts) that implements endpoint as well as SIP proxy functionality. The implementation of this component is virtually complete on the SIP side, but the Mbus interface is continuously being refined as new needs for call control / gatewaying surface.
- H.323 engine – an H.323 Mbus engine providing the same abstract call control interface as the SIP engine does (in addition to H.323 specific services) is under development.
- SAP / SDP engine – the functionality for receiving and interpreting SAP/SDP-based session announcements (including consideration of the SDR cache) is complete. Specification of an abstract Mbus interface for querying / passing information about the session announcements has begun and an implementation of this Mbus interface will follow suit as soon as the specification is complete. This engine will be incorporated in AudioGate as well as in the UCL Transcoding Gateway.
- ISDN call control engine – a basic ISDN call control engine to provide call setup and teardown features as well as a bi-directional DTMF signaling interface is implemented (and is successfully used within AudioGate 0.1). The abstract Mbus interface is likely to be similar to the H.323 and SIP modules' interfaces for call control – as far as the low-level ISDN board drivers provide the necessary information. Extensions for DTMF signaling and channel routing will be included; their specification, however, is currently suspended to allow leveraging the outcome of the IETF's Media Gateway Control (MEGACO) WG's specification on gateway-internal interfaces. Implementation of the Mbus interface for basic call control is nearing completion; advanced services need to wait for the specification to become stable.
- As a possible extension, development of an Mbus engine supporting the Real-Time Streaming Protocol (RTSP) for remote access to media servers along with all the necessary Mbus commands is under consideration within MECCANO.
- The UCL Transcoding Gateway includes a proprietary control protocol, similar in concept to the Real-time Streaming Protocol (RTSP), which is used to control the relaying and transcoding functions of that gateway. This control module currently uses an ad-hoc control protocol, from which the Mbus descended, but it is expected that this will migrate to using the Mbus and RTSP in future.

## **2.3 Media engines**

At present, two media engines developed in MECCANO are designed / extended for use in gateways:

- The Robust-Audio Tool (RAT) is used for media conversion between two different encoding and/or packetization schemes on the IP side. It may also act as an RTP translator – for both IPv4 to IPv6 and unicast to multicast conversion. It has also been extended to provide an interface to an ISDN BRI thereby allowing conversion of audio streams from the line-switched to the packet-switched environment and vice versa.
- Limited video transcoding and multicast-to-unicast conversion is performed by the LBL video gateway (vgw). This is currently controlled via an early conference bus interface, but it is expected that an Mbus interface will be provided in the future – thereby enabling the integration of vgw into the MECCANO architecture.

- A dedicated multicast to unicast packet reflector is also provided.

We expect more than pure audio communication between the packet and the line-switched environments, based upon IP-capable endpoints at both sides with a multicast-unicast gateway in the middle providing the necessary addressing conversion. An optional transcoding gateway can reduce audio/video quality to achieve a transmission rate suitable for the line-switched network.

Full interoperability with H.320-based videoconferencing systems may easily be achieved by combining the MECCANO Mbone-H.323 gateway with an H.323-H.320 gateway. As the latter type of gateway is now commercially available from a variety of manufacturers, no specific efforts are being made within MECCANO to develop such a gateway, although, once we have our Mbone-H.323 gateway in place, we expect to test the complete system with one of these commercial H.320-H.323 based gateways.

### **3. AudioGate**

The MECCANO Audio Mbone-Telephony Gateway “AudioGate” provides users on an arbitrary telephone network (PSTN, ISDN and GSM) with access to the audio channel of Mbone conferences. Upon connection setup, functions such as dynamic conference selection will be provided. As soon as a “connection” to an Mbone session is established, additional services such as user identification, muting, most recent speaker indication, etc. may be provided.

AudioGate is developed in several phases. The initial phase is focusing on implementing, testing, and optionally enhancing the core components with less focus on the overall Mbus architecture. This phase has been completed with the delivery of AudioGate 0.1 that shows all the base technology working. Several further steps are needed to convert AudioGate 0.1 into a gateway based upon the Mbus architecture and to enrich the range of functionality offered to the users.

#### **3.1 AudioGate Version 0.1 Deliverable**

AudioGate version 0.1 is the initial release of an ISDN-Mbone Gateway, jointly developed by TELES and UB TZI. This version is primarily intended as a proof of concept for the following key components: usage of RAT for conversion from line-oriented to packet-oriented audio; use of RAT to realize audio communications with ISDN boards in Linux PCs; integration of this facility into the multimedia communication environment developed by the MECCANO project. In this release, AudioGate is largely based on RAT 3.2.6.

AudioGate provides a dial-in interface that allows users to call a phone number and automatically be transferred into a pre-selected Mbone session. AudioGate uses an ISDN BRI to connect to the phone network and currently allows one voice call (to the same conference). If a Calling Line Identification Presentation (CLIP) service is supported by the caller and the ISDN network, the caller's phone number is provided in the SDES NAME item to identify the person on the phone to the other parties in the Mbone conference.

Version 0.1 of AudioGate is a minimal prototype that does not yet exploit the Mbus as core coordination tool between system components and hence intentionally provides only limited functionality. The focus for this version of AudioGate was the “low level” integration of ISDN call signaling and device handling, understanding the necessary extensions to RAT, and testing the core functionality.

#### **3.2 Target Architecture**

While the version 0.1 release of AudioGate is a single monolithic process, the target architecture splits this process into several different logical components as shown in figure 2. These components are expected to be implemented as separate processes, only the utilization of the same ISDN board for control and data exchange require the ISDN call controller and the RAT engine for a single ISDN B channel to reside within the same process, forming two independent Mbus entities nevertheless.

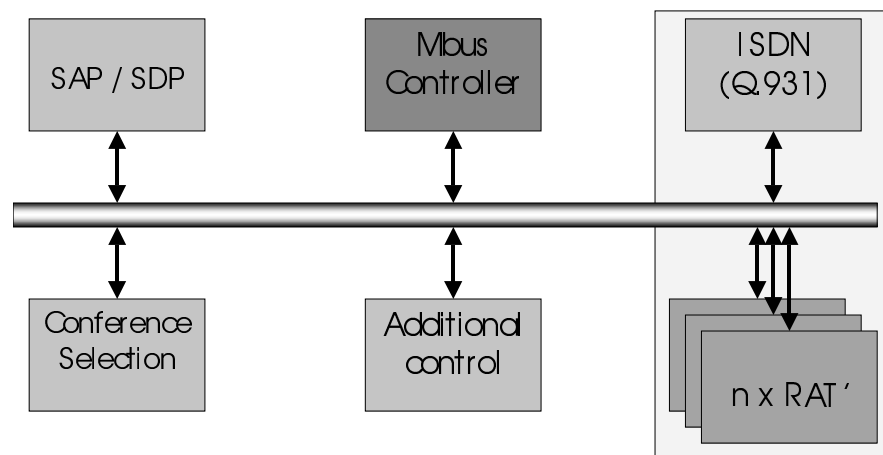


Figure 2: Envisioned Mbus architecture for AudioGate 1.0

The components depicted in figure 2 contribute to the AudioGate functionality as follows:

- The RAT media engine acts as a line switching to packet-switching converter as outlined above.
- The ISDN call controller provides a simple interface to allow setup and teardown of calls, detection of busy and call completion indications and provides an interface towards detection and generation of DTMF signals as well as generation of voice clips for the ISDN side.
- The SAP/SDP module receives session descriptions from the announcement channel(s), extracts the information relevant for identifying and joining Mbone conferences, and provides this information to other Mbus entities in an Mbus-specific format.
- The conference selection module receives session descriptions from the SAP/SDP module and turns them into a choice list assigning each conference a numeric identifier by which the conference to be joined can be picked (via DTMF). The conference selection module also implements a filtering mechanism (to be configured e.g. via a simple resource file) to limit the access to a certain subset of conferences.
- The Mbus entity marked “additional control” is intended to provide an extensible set of additional features, largely based upon DTMF selection by the telephone user. Such services may include muting/un-muting the telephony user, changing the volume, among others.
- Finally, the Mbus controller for the AudioGate glues all these entities together by accepting input from the various other Mbus entities and forwarding them appropriately. For example, directing DTMF input received from the user to the conference selection module while in the setup phase and directing the input to the additional control module when already joined to a conference.

### 3.3 Next steps

As the first step – which is already underway in concept as well as in implementation – this tool will gradually be revised to include all the concepts for Mbus-based gateways developed in the MECCANO system architecture (and thus use the same platform as H.323-SIP, H.323-to-ISDN, or SIP-to-ISDN gateways are supposed to be built upon). The next milestone is to have the same functionality available entirely based upon Mbus components and thus on a gradually extensible platform.

After this, AudioGate will also gain functionality and provide a variety of additional services for the caller from the telephone network. The first services to be addressed include dynamic conference selection and muting/push-to-talk functionality. Additional, more elaborate services will follow which we expect to be controlled through touch-tone signaling. As a pre-requisite, appropriate touch-tone

detection and filtering mechanisms have to be provided, as have facilities for replay and artificial generation of voice output for the ISDN side.

## 4. StarGate

The MECCANO call signaling and media transcoding gateway StarGate is supposed to provide connectivity between different kinds of endpoints interconnected through different types of networks (hence the name “\*Gate”). This is expected to include in particular:

- Conversion between the three most important call signaling protocols (H.323, SIP, and ISDN) including media stream conversion if necessary;
- Actively accessing Mbone sessions from H.323 endpoints; and
- Inviting H.323 endpoints into Mbone sessions for audio and optional video communications.

The architecture of StarGate also allows us to easily extend the number of supported call signaling protocols. In addition, if feasible from the standardization point of view (i.e. the necessary specification are complete and stable), security aspects will be incorporated into the StarGate implementation.

StarGate has not yet been delivered in its initial release but the conceptual work is well progressed, as is the implementation of most of the necessary components. Therefore, this section only addresses the concepts of StarGate while future deliverables will address the further implementation and utilization details. The aforementioned functionality will be included incrementally in releases of StarGate as appropriate.

### 4.1 Mbus Architecture

StarGate is conceptually built upon the same general Mbus architecture as AudioGate, with largely different Mbus entities and different interactions between them, of course. A conceptual outline of a possible StarGate implementation is shown in figure 3. The various components perform the following tasks:

- The H.323, SIP/SDP, and ISDN modules implement call signaling and (as far as applicable) conference control functions for the respective protocol suite.
- An Mbus RAT entity is instantiated whenever transcoding (e.g. for interconnection to the telephone network) is required.
- The Call Routing module provides address and endpoint reachability resolution and, in particular, decides which protocol to route an incoming call across.
- The Access Control module is used to verify that incoming calls are authorized to be completed according to the reachability decision taken by the Call Routing module (e.g. whether an IP-side caller is allowed to call a long-distance number via the telephone network).
- Finally, the Mbus controller again provides the necessary glue between all the modules forwarding call messages back and forth, keeping per call and resource utilization state, etc. In particular, it knows which control protocol entities are present and is optionally capable to translate non-standard Mbus call control messages between the various protocols and instantiate / configure the RAT media engine(s) accordingly.

Further Mbus entities (control applets as well as policy modules) may be introduced to provide additional functionality such as value added services based upon DTMF tones or similar signaling from the IP side.

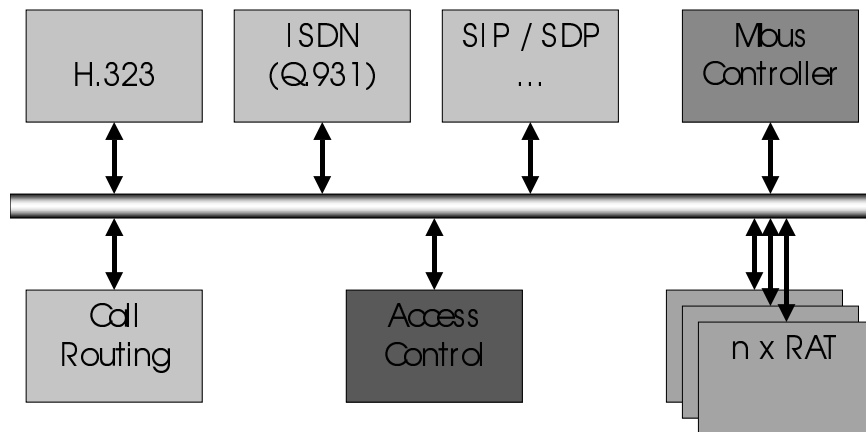


Figure 3: Sample outline of the MECCANO StarGate

All three of the aforementioned control protocol entities share a core set of Mbus messages to set up, tear down, and monitor progress of a call. In addition, each entity supports protocol-specific Mbus extensions that may not be (easily) mapped to other control protocols. The Mbus controller is expected to understand all these Mbus commands, route incoming messages, and optionally perform translation between different protocols. The following section describes the common Mbus commands so far defined for call control.

#### 4.2 Mbus Commands for Call Control

Call control messages are intended for interaction with call control and invitation protocols such as H.323 and SIP. They are designed to constitute the union of the call control messaging needed by endpoints, gateways, proxies, multi-point controllers, and gatekeepers. This allows the use of the Message Bus to act as gluing mechanism to create any type of system from roughly the same building blocks.

Mbus call control messages are based on a common basic message set defined below that will be supported by any kind of call control protocol entity. The basic message set may be augmented by protocol-specific extensions required for protocol specific interactions between a local controller and/or local applications on one side and the respective protocol engine on the other.

A possible future extension for MECCANO is an Mbus command set for the Real-Time Streaming Protocol (RTSP) extensions. However, this is left for further study.

#### Description of Commands

A namespace hierarchy has been defined for the generic and the protocol specific call control commands:

Command prefix	Description
conf.call-control.	basic call control message set
conf.call-control.h323.	extensions for H.323-specific call control messages
conf.call-control.sip.	extensions for SIP-specific call control messages
conf.call-control.isdn.	extensions for ISDN-specific call control messages

The protocol specific extensions are currently being defined, the basic call control command set is presented in the following. H.323, SIP, and ISDN-specific messages are still under study.

## Basic Call Control Commands

The basic set of messages is defined to provide the core functionality of initiating a call on one side, accepting or refusing it on the other, and providing progress information as well as allowing termination of the call on either side. All types of call control engines must support the basic call control message set.

These messages are exchanged using unicast addressing between some local controlling entity and a call control engine implementing a call control or initiation protocol such as H.323 or SIP:

- Outgoing calls may be initiated by any local entity; the call control engine has to keep track of the initiator of a particular call and return all responses or events relating to this call to this entity – which may be different on a per call basis. If the call control engine notices that the controlling entity for a particular call has gone (e.g. because the Mbus reliability mechanism indicates non-delivery of a call control message or a BYE message was seen from this entity), these messages are forwarded to the local controller. If no local controller is available, the call is terminated.
- Indications about incoming calls are always forwarded to the local controller. If no local controller is present the call control engine automatically rejects incoming calls.

All messages of the basic call control message set are sent reliably via unicast to the call control engine.

The basic call control command set contains the messages for establishing a simple (point-to-point) call, along with a few messages dealing with supplementary services.

The following commands are defined:

Command name	Description
conf.call-control.call	The CALL message is sent to the call control engine to make the engine initiate a call to another endpoint using the parameters specified as part of the CALL message.
conf.call-control.disconnect	The DISCONNECT command is sent by the local controller to the call control engine to indicate that the specified call is to be disconnected. It can also be used by the local controller to inform the call control engine that a call has already been terminated by out-of-band communication, e.g. a horizontal conference control protocol. In this case a special reason code has to be passed with the command.
conf.call-control.ringing	The RINGING message is sent from the call control engine to the entity from which it received the corresponding CALL message. RINGING indicates that one or more addresses at the far end were contacted and are now alerting the user.
conf.call-control.connected	The CONNECTED message is sent by the call control engine to the entity that initiated the call (on the calling side) and to the local controller (on the called side) to indicate that the call was successfully established.
conf.call-control.rejected	The REJECTED message is sent by the call control engine to the entity that initiated the call (on the calling side) and to the local controller (on the called side) to indicate that the call was rejected.
conf.call-control.disconnected	The DISCONNECTED message is sent by the call control engine to the entity that initiated the call (on the calling side) and to the local controller (on the called side) to indicate that the call was disconnected.
conf.call-control.incoming-call	The INCOMING CALL message is sent by the call control engine to the local controller to indicate a call request from another endpoint.

Command name	Description
conf.call-control.accept	An ACCEPT message is sent by the local controller to the call control engine that has indicated an INCOMING CALL to indicate acceptance of the call.
conf.call-control.reject	A REJECT message is sent by the local controller to the call control engine that has indicated an INCOMING CALL to indicate rejection of the call.
conf.call-control.redirect	The REDIRECT command is sent by the local controller to the call control engine to indicate that the specified call is to be redirected to another specified address. The third parameter determines whether the call control engine should perform a passive redirection (by telling the caller the redirected address) or an active redirection by operating as a proxy.
conf.call-control.redirected	The REDIRECTED command is sent by a call control engine to the local controller to indicate that the specified call has been redirected to the specified address.
conf.call-control.forward	The FORWARD command is sent by the local controller to the call control engine to indicate that the specified call is to be forwarded to another (optionally specified) address. The FORWARD command can be used instead of REDIRECT when the end system acts as a firewall that decides which calls are to be forwarded. The forwarding can either happen with the call control protocol's implicit semantics (e.g. SIP forwarding) or the controller can explicitly specify the forwarding address.
conf.call-control.forwarded	The FORWARDED command is sent by the call control engine to the local controller to indicate that the specified call has been forwarded to the specified address. The local controller can decide whether the call setup should continue or be interrupted (by sending a DISCONNECT command).

## 5. The UCL Transcoding Gateway

The UCL Transcoding Gateway (UTG) provides access to multicast conferences for hosts with only unicast connectivity. In addition, it provides limited transcoding and mixing functions, primarily for audio.

The initial version of the UTG was developed as part of the MERCI project before the Mbus concept was fully developed. We delivered an enhanced version of this (UTG v1.2) as part of MECCANO deliverable D4.1.

As part of MECCANO we plan to extend the UTG to fully embrace the Mbus architecture in a similar manner to the StarGate system, although the components used are somewhat different. A conceptual outline of the UTG system is illustrated in figure 4.

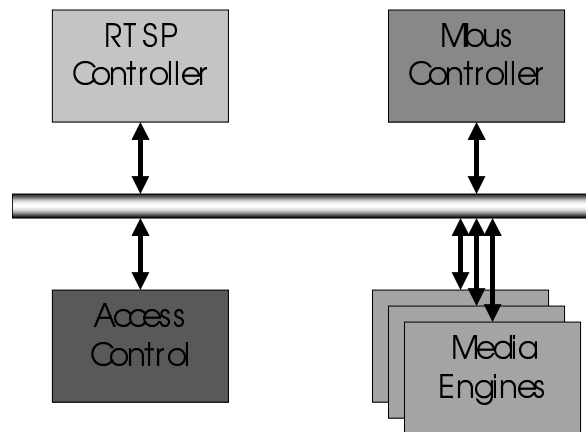


Figure 4: Conceptual outline of the UTG system

The components in the UTG architecture are expected to perform the following tasks:

- The RTSP controller module will provide the control interface to the unicast-only end-system.
- The access control module is used to verify that requests for transcoding and gatewaying are from authorized users.
- One or more media engines to instantiated to perform transcoding and gatewaying when required.
- Finally, the Mbus controller provides the necessary glue between all the other modules.

Although this version of the UTG has not yet been delivered, a number of the necessary components already exist. In particular, the media engines are well developed although some need updating to match the current Mbus specification, rather than earlier ad-hoc control protocols. The access control module is expected to be somewhat similar in concept to that employed in the StarGate system.

The RTSP controller is a new piece of the UTG architecture, and is intended as a replacement for the current control protocol. We expect to develop this in two stages:

1. by integrating an Mbus interface into the current UTG control module, reusing as many of the call control commands defined in section 4.2 of this deliverable as possible; and
2. by converting the current control protocol to use RTSP whilst retaining the control interface.

## 6. Conclusion

The objectives of MECCANO WP6 are to realize several of the forms of gateways and relays that are required: between the ITU-T mechanisms and those used on the Mbone, that can filter and multiplex streams, and that transcode intelligently at network boundaries. By leveraging the Mbus concept developed in WP7 and the media tools developed in WP4 we have provided a solid basis for the development of these gateways.

The work carried out so far in this work package has produced concepts for and implementations of very valuable conferencing / telephony infrastructure components with various immediately obvious areas of application:

- Within the bi-weekly MECCANO management meetings held over the Mbone, various applications are of practical importance:
  - ♣ AudioGate and StarGate allow to bridge in people from anywhere on the PSTN/ISDN/GSM, particularly allowing them to participate in important decisions / discussion while being e.g. on business trips where there may be no, or no reasonable, Mbone or even IP connectivity available.

- ♣ The multicast-to-unicast relays can be used as temporary workaround for the current problems with poor international Mbone connectivity. They will allow us to arrange for temporary unicast relaying of traffic between partner sites when multicast connectivity fails; in addition to connecting those sites that lack multicast connectivity.
- In addition, at the partners' sites various specific applications are envisioned, for example:
  - ♣ At UCL, the UCL Transcoding Gateway is used to allow participation in Mbone sessions from home via dialup PPP links;
  - ♣ At the Universität Bremen, an interconnection of the university's phone system and StarGate is envisaged to circumvent the insufficient number of PBX ports (and hence individual phones). This will allow us to provide direct reachability of all the research group's employees via a combination of Voice over IP applications (based upon the endpoint software delivered in WP4) and conventional telephony.

Further areas of applications are constantly being investigated (not only) at these two sites to simplify (local) collaboration and improve its efficiency, particularly including seamless integration of computer-based communications (web, voice over IP, multimedia conferencing) and conventional communication tools, such as (mobile) telephones.