

BS ISO 17807:2013



BSI Standards Publication

Space data and information transfer systems - Asynchronous message service

NO COPYING WITHOUT BSI PERMISSION EXCEPT AS PERMITTED BY COPYRIGHT LAW

raising standards worldwide™

Full document details: BS ISO 17807:2013



National foreword

This British Standard is the UK implementation of ISO 17807:2013.

The UK participation in its preparation was entrusted to Technical Committee ACE/68/-/7, Space systems and operations - Space data and information transfer systems.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

© The British Standards Institution 2013. Published by BSI Standards Limited 2013

ISBN 978 0 580 77417 1

ICS 49.140

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 30 June 2013.

Amendments issued since publication

Date	Text affected
------	---------------

INTERNATIONAL STANDARD

BS ISO 17807:2013

ISO
17807

First edition
2013-06-01

Space data and information transfer systems — Asynchronous message service

*Systèmes de transfert des informations et données spatiales — Service
de messagerie asynchrone*



Reference number
ISO 17807:2013(E)
Documents
Document Sharing Hub
© ISO 2013



COPYRIGHT PROTECTED DOCUMENT

© ISO 2013

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized otherwise in any form or by any means, electronic or mechanical, including photocopying, or posting on the internet or an intranet, without prior written permission. Permission can be requested from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2. www.iso.org/directives

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received. www.iso.org/patents

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

ISO 17807 was prepared by the Consultative Committee for Space Data Systems (CCSDS) (as CCSDS 735.1-B-1, September 2011) and was adopted (without modifications except those stated in Clause 2 of this International Standard) by Technical Committee ISO/TC 20, *Aircraft and space vehicles*, Subcommittee SC 13, *Space data and information transfer systems*.

Space data and information transfer systems — Asynchronous message service

1 Scope

This International Standard defines a CCSDS Asynchronous Message Service (AMS) for mission data system communications. The service and its protocols implement an architectural concept under which the *modules* of mission systems—distinct sequential flows of application control logic, whether called processes, tasks, or threads—may be designed as if they were to operate in isolation, each one producing and consuming mission information without explicit awareness of which other modules are currently operating. Communication relationships among such modules are self-configuring; this tends to minimize complexity in the development and operations of modular data systems.

A system built on this model is a ‘society’ of generally autonomous interoperating modules that may fluctuate freely over time in response to changing mission objectives, module functional upgrades, and recovery from individual module failure. The purpose of AMS, then, is to reduce mission cost and risk by providing standard, reusable infrastructure for the exchange of information among data system modules in a manner that is simple to use, highly automated, flexible, robust, scalable, and efficient.

This International Standard specifies the protocol procedures and data units that accomplish automatic configuration of AMS communication relationships, dynamic reconfiguration of those relationships during operations, and the use of those relationships to accomplish the exchange of mission information among data system modules.

The scope and field of application are furthermore detailed in subclause 1.2 of the enclosed CCSDS publication.

2 Requirements

Requirements are the technical recommendations made in the following publication (reproduced on the following pages), which is adopted as an International Standard:

CCSDS 735.1-B-1, September 2011, Asynchronous message service.

For the purposes of international standardization, the modifications outlined below shall apply to the specific clauses and paragraphs of publication CCSDS 735.1-B-1.

Pages i to vi

This part is information which is relevant to the CCSDS publication only.

Page 1-8

Add the following information to the reference indicated:

[3] Document CCSDS 301.0-B-4, November 2010, is equivalent to ISO 11104:2013.

3 Revision of publication CCSDS 735.1-B-1

It has been agreed with the Consultative Committee for Space Data Systems that Subcommittee ISO/TC 20/SC 13 will be consulted in the event of any revision or amendment of publication CCSDS 735.1-B-1. To this end, NASA will act as a liaison body between CCSDS and ISO.

Recommendation for Space Data System Standards

ASYNCHRONOUS MESSAGE SERVICE

RECOMMENDED STANDARD

CCSDS 735.1-B-1

BLUE BOOK
September 2011

(Blank page)

AUTHORITY

Issue:	Recommended Standard, Issue 1
Date:	September 2011
Location:	Washington, DC, USA

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS documents is detailed in the *Procedures Manual for the Consultative Committee for Space Data Systems*, and the record of Agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the address below.

This document is published and maintained by:

CCSDS Secretariat
Space Communications and Navigation Office, 7L70
Space Operations Mission Directorate
NASA Headquarters
Washington, DC 20546-0001, USA

STATEMENT OF INTENT

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of its members. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommended Standards** and are not considered binding on any Agency.

This **Recommended Standard** is issued by, and represents the consensus of, the CCSDS members. Endorsement of this **Recommendation** is entirely voluntary. Endorsement, however, indicates the following understandings:

- o Whenever a member establishes a CCSDS-related **standard**, this **standard** will be in accord with the relevant **Recommended Standard**. Establishing such a **standard** does not preclude other provisions which a member may develop.
- o Whenever a member establishes a CCSDS-related **standard**, that member will provide other CCSDS members with the following information:
 - The **standard** itself.
 - The anticipated date of initial operational capability.
 - The anticipated duration of operational service.
- o Specific service arrangements shall be made via memoranda of agreement. Neither this **Recommended Standard** nor any ensuing **standard** is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this **Recommended Standard** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or (3) be retired or canceled.

In those instances when a new version of a **Recommended Standard** is issued, existing CCSDS-related member standards and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each member to determine when such standards or implementations are to be modified. Each member is, however, strongly encouraged to direct planning for its new standards and implementations towards the later version of the Recommended Standard.

FOREWORD

As the computers used to conduct space flight mission operations both in flight and on the ground increase in capability, the software operating on those computers tends to increase in functional scope and thus take on greater operational significance. With that increase in scope comes increasing size and complexity, which may be partially mitigated by decomposition into modules whose functionality can be readily defined and tested. However, this modularity in turn entails a growing reliance on effective communication among modules. That is, mere decomposition cannot diminish the functional complexity implied by complex requirements. It can only partition that complexity into manageable portions, while the resulting web of communication relationships among modules introduces new complexity of a different order: rather than a relatively simple system of a few increasingly large and complex modules, a modern mission is increasingly likely to require a large and complex system of relatively simple modules.

Increasing complexity tends to increase the likelihood of failure. The increasing complexity of mission systems based on communication among modules therefore tends to increase the chance of such systems' failing even as the success of those systems becomes increasingly critical to the achievement of mission objectives. Measures that can minimize the chance of failure in complex systems—exhaustive regression testing and configuration management, flight rules constraining the exercise of unproven system capabilities and the introduction of improvements—increase cost if they are taken and increase risk if they are not.

These considerations have led to the present recommendation for a standard system of communication—*messaging*—among mission software modules. The objective of this proposed messaging standard is to reduce mission cost and risk by confining much of the complexity of modern mission systems to relatively static and proven reusable infrastructure.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CCSDS shall not be held responsible for identifying any or all such patent rights.

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommended Standard is therefore subject to CCSDS document management and change control procedures, which are defined in the *Procedures Manual for the Consultative Committee for Space Data Systems*. Current versions of CCSDS documents are maintained at the CCSDS Web site:

<http://www.ccsds.org/>

Questions relating to the contents or status of this document should be addressed to the CCSDS Secretariat at the address indicated on page i.

At time of publication, the active Member and Observer Agencies of the CCSDS were:

Member Agencies

- Agenzia Spaziale Italiana (ASI)/Italy.
- Canadian Space Agency (CSA)/Canada.
- Centre National d'Etudes Spatiales (CNES)/France.
- China National Space Administration (CNSA)/People's Republic of China.
- Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)/Germany.
- European Space Agency (ESA)/Europe.
- Federal Space Agency (FSA)/Russian Federation.
- Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
- Japan Aerospace Exploration Agency (JAXA)/Japan.
- National Aeronautics and Space Administration (NASA)/USA.
- UK Space Agency/United Kingdom.

Observer Agencies

- Austrian Space Agency (ASA)/Austria.
- Belgian Federal Science Policy Office (BFSPO)/Belgium.
- Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
- China Satellite Launch and Tracking Control General, Beijing Institute of Tracking and Telecommunications Technology (CLTC/BITTT)/China.
- Chinese Academy of Sciences (CAS)/China.
- Chinese Academy of Space Technology (CAST)/China.
- Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
- CSIR Satellite Applications Centre (CSIR)/Republic of South Africa.
- Danish National Space Center (DNSC)/Denmark.
- Departamento de Ciência e Tecnologia Aeroespacial (DCTA)/Brazil.
- European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
- European Telecommunications Satellite Organization (EUTELSAT)/Europe.
- Geo-Informatics and Space Technology Development Agency (GISTDA)/Thailand.
- Hellenic National Space Committee (HNSC)/Greece.
- Indian Space Research Organization (ISRO)/India.
- Institute of Space Research (IKI)/Russian Federation.
- KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
- Korea Aerospace Research Institute (KARI)/Korea.
- Ministry of Communications (MOC)/Israel.
- National Institute of Information and Communications Technology (NICT)/Japan.
- National Oceanic and Atmospheric Administration (NOAA)/USA.
- National Space Agency of the Republic of Kazakhstan (NSARK)/Kazakhstan.
- National Space Organization (NSPO)/Chinese Taipei.
- Naval Center for Space Technology (NCST)/USA.
- Scientific and Technological Research Council of Turkey (TUBITAK)/Turkey.
- Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
- Swedish Space Corporation (SSC)/Sweden.

- United States Geological Survey (USGS)/USA.

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 735.1-B-1	Asynchronous Message Service, Recommended Standard, Issue 1	September 2011	Original issue

CONTENTS

<u>Section</u>	<u>Page</u>
1 INTRODUCTION	1-1
1.1 PURPOSE AND SCOPE.....	1-1
1.2 APPLICABILITY	1-1
1.3 CONVENTIONS AND DEFINITIONS	1-2
1.4 REFERENCES	1-8
2 OVERVIEW	2-1
2.1 GENERAL.....	2-1
2.2 ARCHITECTURAL ELEMENTS	2-16
2.3 OVERVIEW OF INTERACTIONS	2-17
3 SERVICE DESCRIPTIONS	3-1
3.1 SERVICES PROVIDED TO THE APPLICATION	3-1
3.2 SERVICES REQUIRED OF THE TRANSPORT SERVICE.....	3-26
4 PROTOCOL SPECIFICATION	4-1
4.1 GENERAL.....	4-1
4.2 META-AMS CONFIGURATION PROCEDURES	4-2
4.3 APPLICATION AMS COMMUNICATION PROCEDURES	4-14
4.4 REMOTE AMS GATEWAY PROCEDURES	4-19
5 PROTOCOL DATA UNITS	5-1
5.1 META-AMS PROTOCOL DATA UNITS	5-1
5.2 AAMS PROTOCOL DATA UNITS	5-10
5.3 REMOTE AMS PROTOCOL DATA UNITS	5-12
5.4 IMPLEMENTATION NOTE	5-14
6 USER OPERATIONS	6-1
6.1 AMS PROXY ARCHITECTURE OVERVIEW.....	6-1
6.2 AMQP-BASED AMSP.....	6-2
7 CONFORMANCE CLASSES	7-1
7.1 OVERVIEW	7-1
7.2 CAPABILITY SETS.....	7-1

7.3	PROCEDURES INCLUDED IN CAPABILITY SETS	7-2
-----	--	-----

CONTENTS (continued)

<u>Section</u>	<u>Page</u>
8 MANAGEMENT INFORMATION BASE	8-1
8.1 MODULE MIB	8-1
8.2 REGISTRAR MIB	8-2
8.3 CONFIGURATION SERVER MIB	8-3
ANNEX A RECOGNIZED TRANSPORT SERVICES (NORMATIVE)	A-1
ANNEX B RESERVED SERVICE PARAMETER VALUES (NORMATIVE)	B-1
ANNEX C SERVICE CONFORMANCE STATEMENT PROFORMA (NORMATIVE)	C-1
ANNEX D PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT PROFORMA (NORMATIVE)	D-1
ANNEX E SECURITY (INFORMATIVE)	E-1
ANNEX F INFORMATIVE REFERENCES (INFORMATIVE)	F-1
ANNEX G ACRONYMS (INFORMATIVE)	G-1

Figure

1-1 Bit Numbering Convention	1-2
1-2 Octet Convention	1-3
2-1 Architectural Elements of AMS (a Single Continuum)	2-16
2-2 General Remote AMS Architecture	2-25

Table

1-1 AMS Timeout Intervals (in Seconds)	1-8
5-1 MAMS PDU Header Fields	5-1
5-2 MAMS PDU Types	5-2
5-3a MAMS PDU Structures (Sorted by Type)	5-9
5-3b MAMS PDU Structures (Sorted Alphabetically)	5-10
5-4 AAMS Message Header Fields	5-11
5-5 Envelope Header Fields	5-13
5-6 Envelope Header Field Contents	5-13
7-1 Capability Sets Encompassed by Conformance Classes	7-2

(Blank page)

1 INTRODUCTION

1.1 PURPOSE AND SCOPE

This document defines a CCSDS Asynchronous Message Service (AMS) for mission data system communications. The service and its protocols implement an architectural concept under which the *modules* of mission systems—distinct sequential flows of application control logic, whether called processes, tasks, or threads—may be designed as if they were to operate in isolation, each one producing and consuming mission information without explicit awareness of which other modules are currently operating. Communication relationships among such modules are self-configuring; this tends to minimize complexity in the development and operations of modular data systems.

A system built on this model is a ‘society’ of generally autonomous interoperating modules that may fluctuate freely over time in response to changing mission objectives, module functional upgrades, and recovery from individual module failure. The purpose of AMS, then, is to reduce mission cost and risk by providing standard, reusable infrastructure for the exchange of information among data system modules in a manner that is simple to use, highly automated, flexible, robust, scalable, and efficient.

This Recommended Standard specifies the protocol procedures and data units that accomplish automatic configuration of AMS communication relationships, dynamic reconfiguration of those relationships during operations, and the use of those relationships to accomplish the exchange of mission information among data system modules.

1.2 APPLICABILITY

This Recommended Standard specifies protocols and associated services that enable communication among modules of mission data systems, specifically:

- between modules of a ground data system;
- between modules of a flight data system;
- between modules of different ground data systems;
- between modules of the flight data systems of different spacecraft;
- between modules of flight data systems and modules of ground data systems, over interplanetary distances.

1.2.1 ORGANIZATION OF THE RECOMMENDED STANDARD

This Recommended Standard is organized as follows:

- Section 2 provides an overview of AMS, its intended use, and a description of the main interactions involved in message exchange.

- Section 3 defines the services provided by AMS along with the associated primitives and parameters.
- Section 4 defines the AMS protocol procedures.
- Section 5 defines the AMS protocol data units.
- Section 6 defines standardized user operations that may be built on AMS.
- Section 7 defines eight Conformance Classes of AMS implementation.
- Section 8 describes the AMS Management Information Base (MIB).
- Annex A discusses the currently recognized underlying transport services for AMS.
- Annex B summarizes the functional significance of various classes of AMS subject numbers and the manner in which this significance affects protocol procedures.
- Annex C defines a Service Conformance Statement Proforma.
- Annex D defines a Protocol Implementation Conformance Statement (PIC) Proforma.
- Annex E discusses security, SANA, and patent considerations related to the specification.
- Annex F provides a list of informative references.
- Annex G provides a list of acronyms and definitions.

1.3 CONVENTIONS AND DEFINITIONS

1.3.1 BIT NUMBERING CONVENTION

In this document, the following convention is used to identify each bit in an N -bit field. The first bit in the field to be transmitted (i.e., the most left justified when drawing a figure) is defined to be 'Bit 0'; the following bit is defined to be 'Bit 1' and so on up to 'Bit $N-1$ '. When the field is used to express a binary value (such as a counter), the Most Significant Bit (MSB) shall be the first transmitted bit of the field, i.e., 'Bit 0'.

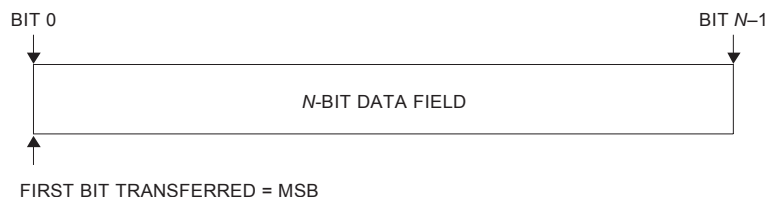


Figure 1-1: Bit Numbering Convention

In accordance with modern data communications practice, spacecraft data fields are often

grouped into eight-bit ‘words’ which conform to the above convention. Throughout this Recommended Standard, the following nomenclature is used to describe this grouping:

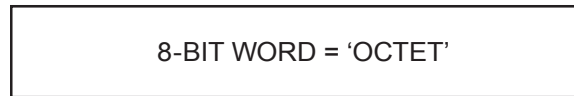


Figure 1-2: Octet Convention

By CCSDS convention, all ‘spare’ or ‘unused’ bits shall be permanently set to value ‘zero’.

1.3.2 NOMENCLATURE

The following conventions apply for the normative specifications in this Recommended Standard:

- a) the words ‘shall’ and ‘must’ imply a binding and verifiable specification;
- b) the word ‘should’ implies an optional, but desirable, specification;
- c) the word ‘may’ implies an optional specification;
- d) the words ‘is’, ‘are’, and ‘will’ imply statements of fact.

NOTE – These conventions do not imply constraints on diction in text that is clearly informative in nature.

1.3.3 INFORMATIVE TEXT

In the normative sections of this document, sections 3 through 8 and annexes A through D, informative text is set off from the normative specifications either in notes or under one of the following subsection headings:

- Overview;
- Background;
- Rationale;
- Discussion.

1.3.4 DEFINITIONS

1.3.4.1 Definitions from OSI Basic Reference Model

This Recommended Standard makes use of a number of terms defined in reference [1]. The use of those terms in this Recommended Standard shall be understood in a generic sense, i.e., in the sense that those terms are generally applicable to any of a variety of technologies that provide for the exchange of information between real systems. Those terms are:

- entity;
- Protocol Data Unit (PDU);
- service;
- Service Access Point (SAP);
- Service Data Unit (SDU).

1.3.4.2 Definitions from Open Systems Interconnection (OSI) Service Definition Conventions

This Recommended Standard makes use of a number of terms defined in reference [2]. The use of those terms in this Recommended Standard shall be understood in a generic sense, i.e., in the sense that those terms are generally applicable to any of a variety of technologies that provide for the exchange of information between real systems. Those terms are:

- Indication;
- Primitive;
- Request;
- Response.

1.3.4.3 Terms Defined in This Recommended Standard

NOTE – In view of the detailed technical rigor of these definitions, the first-time reader is encouraged to skip them and instead begin at the much more accessible section 2, below, referring back to the definitions in 1.3.4.3 as necessary.

Within the context of this document the following definitions apply:

continuum: A closed set of entities that utilize the Meta-AMS protocol (discussed below) for the purpose of configuring AMS-based communication among themselves. Each continuum is identified by a *continuum name* and corresponding non-negative continuum number. A ‘continuum ID’ is either a continuum name or a continuum number, whichever is more appropriate in context.

application: A data system implementation, typically taking the form of a set of source code text files, that relies on AMS procedures to accomplish its purposes. Each application is identified by an *application name*.

authority: An administrative entity or persona that may have responsibility for the configuration and operation of an instance of an application. Each authority is identified by an *authority name*.

venture: An instance of an application, i.e., a functioning projection of the application, for which some authority is responsible, onto a set of one or more running computers. As such it is identified by the combination of some application name and some authority name, and it is alternatively identified by some non-negative, non-zero *venture number* that corresponds to that combination of application and authority names. A ‘venture ID’ is either an application-name/authority-name combination or a venture number, whichever is more appropriate in context

message: An octet array of known size which, when copied from the memory of one module of a venture to that of another (*exchanged*), conveys information that can further the purposes of that venture.

application data: The array of zero or more octets embedded in a message containing the specific information that the message conveys.

role: Some part of the functionality of an application. Each role is identified by a *role name* and corresponding non-negative role number. A ‘role ID’ is either a role name or a role number, whichever is more appropriate in context.

module: A communicating entity that implements some part of the functionality of some AMS venture—that is, performs some application role—by, among other activities, exchanging messages with other modules. Associated with each module is the name of the role it performs within the application. (Multiple modules may perform the same role in an application, so the role name of a module need not uniquely identify the module within its venture.) In order to accomplish AMS message exchange a module generates AMS service requests and consumes AMS service indications; the module that is the origin of a given AMS service request or the destination of a given AMS service indication is termed *the operative module*.

message space: The set of all of the modules of one AMS venture that are also members of a given AMS continuum; that is, a message space is the intersection of a venture and a continuum. Each message space is uniquely identified within its continuum by the combination of the name of the application and the name of the authority that is responsible for the venture, and by the corresponding venture number.

NOTE – Unique naming of continua enables multiple message spaces that are in different continua but are identified by the same application and authority names to be concatenated via Remote AMS (discussed below) into a single venture.

unit (i.e., a unit of organization): An identified subset of the organizational hierarchy of the modules of one AMS venture, declared during venture configuration as specified by the responsible authority for that venture. Each unit is uniquely identified within the venture by *unit name* and corresponding non-negative unit number. A ‘unit ID’ is either a unit name or a unit number, whichever is more appropriate in context. The *root unit* of a venture is the unit that is coterminous with the venture itself; its unit name is the character string that is of length zero. A unit whose name is identical to the first N bytes (where N is greater than or equal to zero) of the name of another unit of the same venture is said to *contain* that other unit. The membership of a unit that is contained by another unit is a subset of the membership of the containing unit. Units form a strict hierarchy: no module is ever a member of two units such that neither unit is a proper subset of the other.

cell: The set of all modules that are members of one unit of a given venture and are also members of a given continuum; that is, it is the intersection of a unit and a continuum. Since each unit is a subset of a venture, each cell is necessarily a subset of the message space for that venture in that continuum. Each cell is uniquely identified within its message space by its unit’s name and number. The root cell of a message space is coterminous with the message space itself. A cell contains some other cell only if its unit contains that other cell’s unit. A cell may be an empty set; that is, in a given continuum there may be no modules that are members of the cell’s unit. The *registered membership* of a cell is the set of all modules in the cell that are not members of any other cell aside from cells which contain that cell.

NOTES

- 1 For example, if cell A contains cells B and C, and cell C contains cells D and E, any modules in C that are not in either D or E are in the registered membership of cell C. Those modules are also members of cell A, but because they are in cell C—which does not contain cell A—they are not in cell A’s registered membership.
- 2 The root cell contains every other cell in the message space, and every module in the message space is therefore a member, though not necessarily a registered member, of the root cell.

domain (of an AMS service request): The set of modules to which the request pertains. It comprises all of the modules that are members of the venture in which the operative module is itself a member, with the following exceptions:

- If the service request is one for which continuum ID is not specified, then only modules that are members of the local continuum are members of the service request’s domain. Otherwise, if the service request’s continuum ID parameter does not indicate ‘all continua’, then only modules that are members of the continuum identified by the service request’s continuum ID parameter are members of the service request’s domain.
- Only modules that are members of the organizational unit identified by the service request’s unit ID parameter are members of the service request’s domain.

- If the service request's role ID parameter does not indicate 'all roles', then only modules performing the role identified by that role ID are members of the service request's domain.

subject number or **subject** (of a message): An integer embedded in the message that indicates the general nature of the information the message conveys, in the context of the AMS venture within which the message is exchanged.

subject name: A text string that serves as the symbolic representation of some subject number. A 'subject ID' is either a subject number or the corresponding subject name, whichever is more appropriate in context.

send: To cause a message to be copied to the memory of a specified module.

publish: To cause a message on a specified subject to be sent to one or more implicitly specified modules, namely, all those that have requested copies of all messages on the specified subject.

announce: To cause a message to be sent to one or more implicitly specified modules, namely, all those modules that are located within a specified continuum (or all continua), that are members of a specified unit (possibly the root unit), and that perform a specified role in the application (possibly 'any role').

subscribe: To issue a *subscription* on a subject, i.e., a request that one copy of every message published on some specified subject by any module in the subscription's *domain* be sent to the subscribing module; the domain of a subscription is the domain of the AMS service request that established the subscription.

invitation: A statement of the manner in which messages on some specified subject may be sent to the inviting module by modules in the *domain* of the invitation; the invitation's domain is the domain of the AMS service request that established the invitation.

reply: A message that is sent back to the sender of some earlier message and that directly responds to that antecedent message in some application-specific way.

context number: A numeric value that (in some application-specific way) enables the information in a reply message to be applied in the context in which the antecedent message was sent. For this purpose, the context number, if any, included in the antecedent message is simply echoed in the reply message.

query: A message that implicitly notifies the receiver that the sender of the message has suspended message exchange altogether until it has received a reply that responds to this message.

Some of the protocol procedures defined in this specification are in part expressed in terms of timeout intervals. All AMS timeout intervals are notional rather than fixed. (However, effective operation of the AMS protocols cannot be assured unless the same fixed values for

all timeout intervals are used universally within any given AMS continuum.) Similarly, the system health monitoring procedures defined in this specification are in part expressed in terms of the number of consecutive missing heartbeats that will be interpreted as indicating that the source entity for the missing heartbeats has ceased operation. This limit is again notional rather than fixed; it is identified in this specification by the label 'N6', and its nominal value is 3.

The timeout intervals referenced in this specification are defined as follows:

Table 1-1: AMS Timeout Intervals (in Seconds)

Interval Name	Interval ID	Nominal Value	Mandatory value
Configuration server response	N1	5.0	n/a
Registrar response	N2	5.0	n/a
Registrar heartbeat	N3	10.0	n/a
Module heartbeat	N4	n/a	2 X the value of N3
Module loss imputation	N5	n/a	N6 X the value of N4

NOTE – An example of an AMS-based communications configuration that illustrates some of the concepts defined above is provided in 2.1.3.

1.4 REFERENCES

The following documents contain provisions which, through reference in this text, constitute provisions of this Recommended Standard. At the time of publication, the editions indicated were valid. All documents are subject to revision, and users of this Recommended Standard are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS documents.

- [1] *Information Technology—Open Systems Interconnection—Basic Reference Model: The Basic Model*. International Standard, ISO/IEC 7498-1:1994. Geneva: ISO, 1994.
- [2] *Information Technology—Open Systems Interconnection—Basic Reference Model—Conventions for the Definition of OSI Services*. International Standard, ISO/IEC 10731:1994. Geneva: ISO, 1994.
- [3] *Time Code Formats*. Recommendation for Space Data System Standards, CCSDS 301.0-B-4. Blue Book. Issue 4. Washington, D.C.: CCSDS, November 2010.
- [4] *AMQP—Advanced Message Queuing Protocol—Protocol Specification*. Version 0-10. N.p.: AMQP Working Group, 2009. <<http://amqp.org/>>

NOTE – Informative references are contained in annex F.

2 OVERVIEW

2.1 GENERAL

2.1.1 ARCHITECTURAL CHARACTER

A data system based on AMS has the following characteristics:

- Any module may be introduced into the system at any time. That is, the order in which system modules commence operation is immaterial; a module never needs to establish an explicit *a priori* communication ‘connection’ or ‘channel’ to any other module in order to pass messages to it or receive messages from it.
- Any module may be removed from the system at any time without inhibiting the ability of any other module to continue sending and receiving messages. That is, the termination of any module, whether planned or unplanned, only causes the termination of other modules that have been specifically designed to terminate in this event.
- When a module must be upgraded to an improved version, it may be terminated and its replacement may be started at any time; there is no need to interrupt operations of the system as a whole.
- When the system as a whole must terminate, the order in which the system’s modules cease operation is immaterial.

AMS-based systems are highly robust, lacking any innate single point of failure and tolerant of unplanned module termination. At the same time, communication within an AMS-based system can be rapid and efficient:

- Messages are exchanged directly between modules rather than through any central message dispatching nexus.
- Messages are automatically conveyed using the most suitable underlying transport service to which the sending and receiving modules both have access. For example, messages between two ground system modules running in different computers on a common LAN would likely be conveyed via TCP/IP, while messages between modules running on two flight processors connected to a common bus memory board might be conveyed via a shared-memory message queue.

Finally, AMS is designed to be highly scalable: partitioning message spaces into cells enables a venture to comprise hundreds or thousands of cooperating modules without significant impact on application performance.

2.1.2 MESSAGE EXCHANGE MODELS

AMS is designed to support a variety of message exchange models:

- *Send/receive.* Messages may simply be sent to designated modules in an asynchronous manner.
- *Synchronous query.* Messages may be sent in a synchronous manner, i.e., suspending further activity until a reply is received.
- *Publish/subscribe.* Messages may be published anonymously to a time-varying community of self-selected subscribers.
- *Announcement.* Messages may be ‘announced’ to a set of modules selected by the message source.

To this end, AMS comprises three distinct and complementary protocols:

- The *Application AMS (AAMS)* protocol conveys application data between AMS modules within a single continuum.
- The *Meta-AMS (MAMS)* protocol propagates configuration information that enables the exchange of AAMS protocol data units within a single continuum.
- The *Remote AMS (RAMS)* protocol conveys application data between AMS modules in different continua.

AAMS message exchange is fundamentally *asynchronous*. That is, each message is sent in a ‘postal’ rather than ‘telephonic’ manner: upon sending a message, an AMS module need not wait for arrival of any message (such as a reply to the message it sent) before continuing performance of its functions.

Although message exchange among modules is asynchronous, for some purposes it may be desirable to apply the information in a reply message (received asynchronously) to the context in which the antecedent message was published. To this end, AAMS enables a module to include a context number in any original (non-reply) message; AAMS procedures can be used to reply to any original message, and the reply to a message automatically includes an echo of the context number (if any) embedded in the original message. This number can be used to retrieve some block of contextual information, enabling the original message sender to link the information in a reply message to the application activity which caused the antecedent message to be issued, so that this activity may be continued in a pseudo-synchronous fashion. The specific mechanism used to establish this linkage is an implementation matter.

For some purposes true message synchrony may be necessary as well; that is, it may be desirable for a module to query some other module, sending a message and then suspending message exchange altogether until a reply is received. AAMS procedures additionally support this communication model when it is required.

Most message exchange in an AMS-based data system is conducted on the ‘publish-subscribe’ model:

- A module uses MAMS procedures to announce that it is subscribing to messages on a specified subject.
- From that time on (until the subscription is canceled), whenever any module in the message space uses AAMS procedures to publish a message on that subject, a copy of the message is automatically delivered to that subscribing module and to all others that have announced similar subscriptions.

This model can greatly simplify application development and integration. In effect, each module plugs itself into a data ‘grid’, much as producers and consumers of electric power—for example, a hydroelectric plant and a kitchen toaster—plug into an electric power grid. An AMS module can insert into such a data grid whatever data it produces, without having to know much about the consumer(s) of that data, and draw from the grid whatever data it requires without having to know much about the producer(s). The design of a module is largely decoupled from the designs of all other modules in the same way that the design of a toaster is largely decoupled from the design of a power plant.

For some purposes, though, it may still be necessary for a module to send a message privately and explicitly to some specific module, e.g., in reply to a published message. Again, AAMS procedures support this communication model as well when it is required.

Finally, a module may also announce a message privately to some explicitly specified set of modules (thus privately and implicitly to all modules in that set), where the specification of the set may comprise any combination of continuum identifier, unit identifier, and role identifier. The announcement model enables the source of the message to exercise some discretion in multi-point data distribution, directing the message only to the modules it believes need to receive it, while still shielding destination modules from the delivery of messages they prefer not to receive.

2.1.3 AMS COMMUNICATION CONFIGURATION EXAMPLE

In this example, AMS is being used to operate a rover on a distant planet. This application is named ‘rover-ops’.

For this purpose, AMS must be running in two distinct continua: ‘rover’, sited on the rover’s flight computer, and ‘moc’, sited on the rover’s Mission Operations Center (MOC) computers on Earth.

NOTE – It is not possible to run the entire system in a single continuum because the Meta-AMS (MAMS) protocol relies on timely responses to messages: the lengthy signal propagation delays of interplanetary communication would prevent the successful completion of MAMS message exchanges. The Remote AMS protocol propagates the results of MAMS-based continuum configuration among continua in a delay-tolerant manner.

It is desirable to run this system both ‘live’ and in simulation, without modifying the software in any way, so two different authorities, named ‘live’ and ‘sim’, are identified.

Finally, it is desirable to be able to send a request for current thermal status information to all modules of the thermal subsystem on the rover and no other modules. For this purpose it is assumed that one unit is defined for each rover-ops venture will be the ‘thermal’ unit.

Figure 2-1 depicts one of the two continua, the **rover** continuum. Each continuum has a single configuration server (discussed later), marked ‘C’ in this diagram.



Figure 2-1: The Rover Continuum

In figure 2-2, the **moc** continuum is depicted as well.

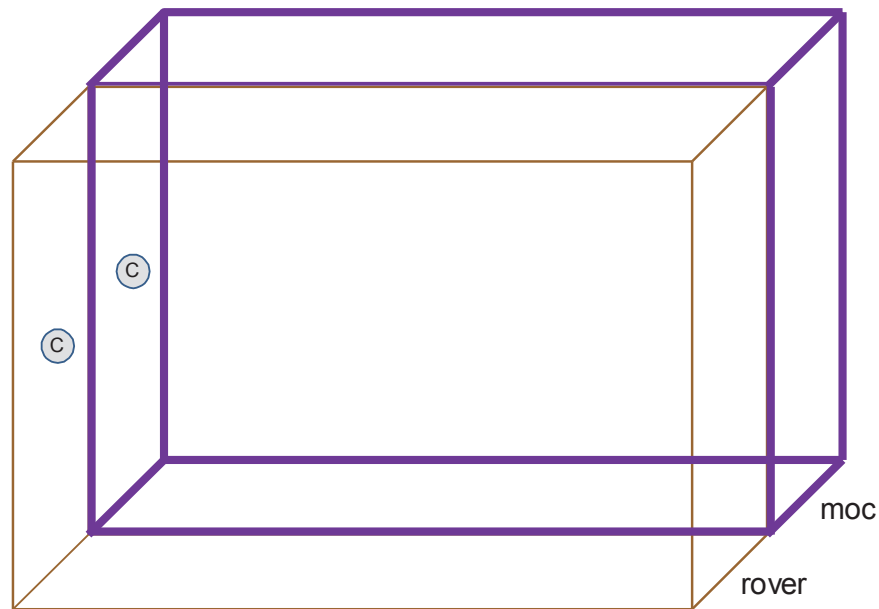


Figure 2-2: Rover and MOC Continua

Figure 2-3 shows a single venture that conducts the **live** operations of the **rover-ops** application, spanning these two continua. Every venture comprises at least one implicit unit, the **root** unit, but since the root unit is coterminous with the venture it is not shown separately here.

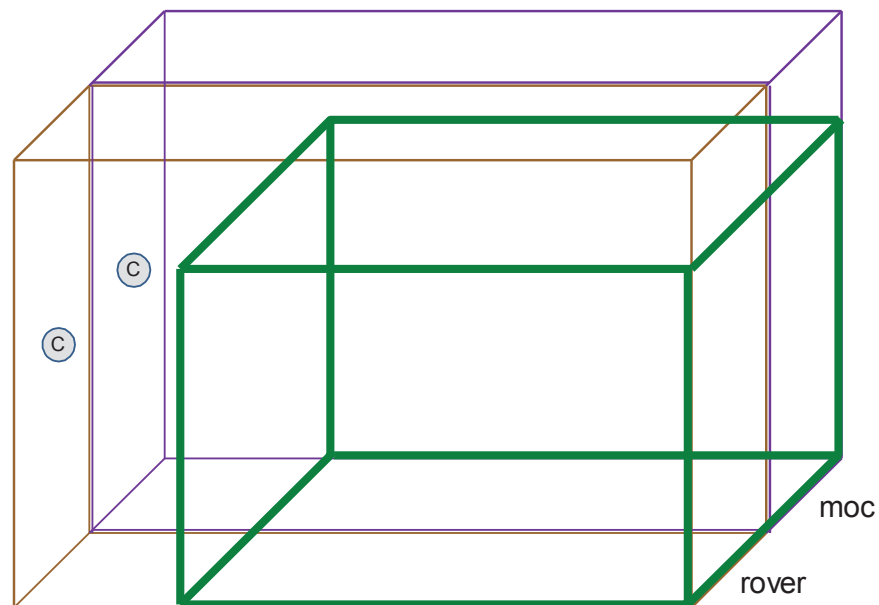


Figure 2-3: Live Rover-Ops Venture

The portion of the **live rover-ops** venture that resides in the **rover** continuum is a single message space as shown in figure 2-4.

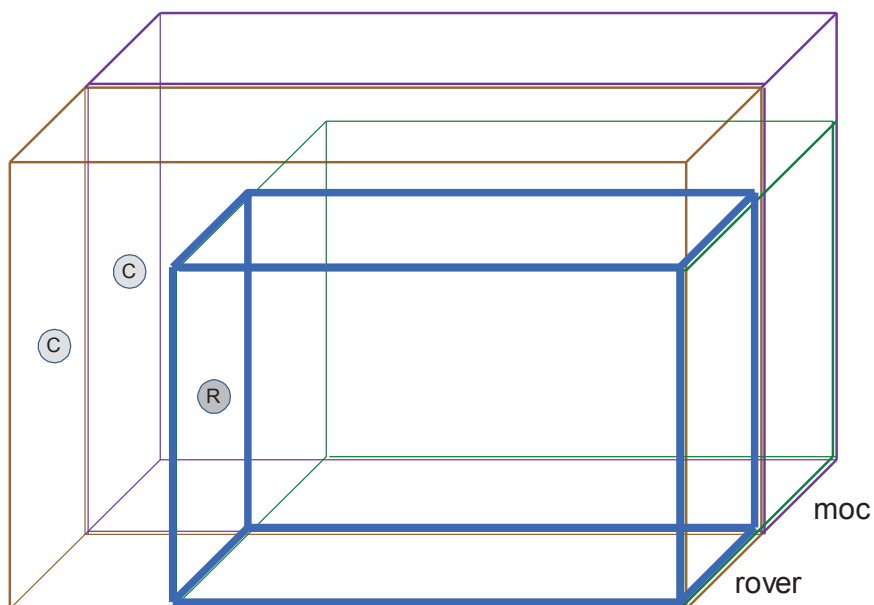


Figure 2-4: Rover Continuum Portion of Live Rover-Ops Venture

Each message space comprises at least one cell, the **root** cell, which is that portion of the **live rover-ops root** unit that resides in the **rover** continuum. Since the **root** cell is coterminous with the message space itself it is not shown separately. Every cell of every message space, including the **root** cell, has a single registrar (discussed later), marked 'R' in this diagram.

In figure 2-5, the message space for the **MOC** portion of **live rover-ops** is added.

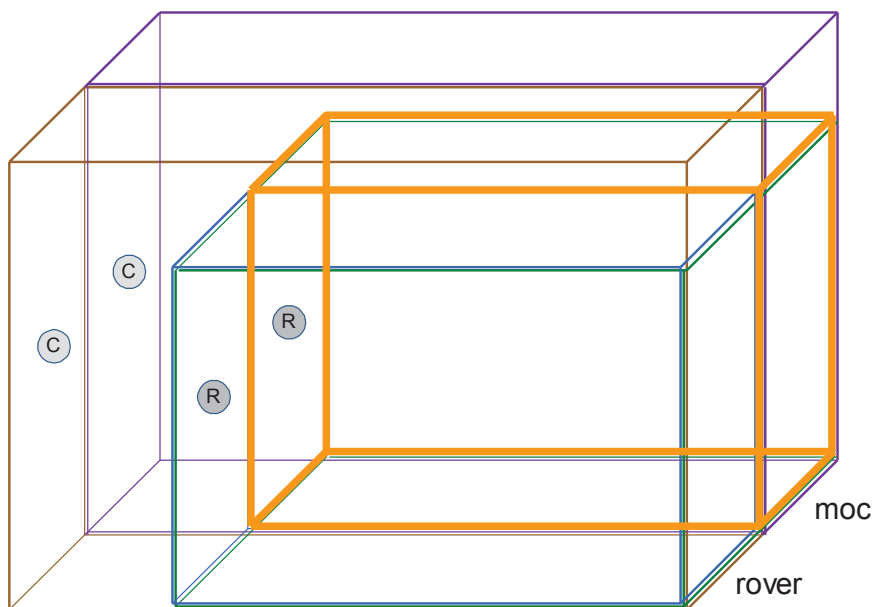


Figure 2-5: Live Rover-Ops Venture with MOC Message Space

Figure 2-6 shows a subset of the **live rover-ops** venture, the **thermal** unit, which spans the **rover** and **moc** continua.

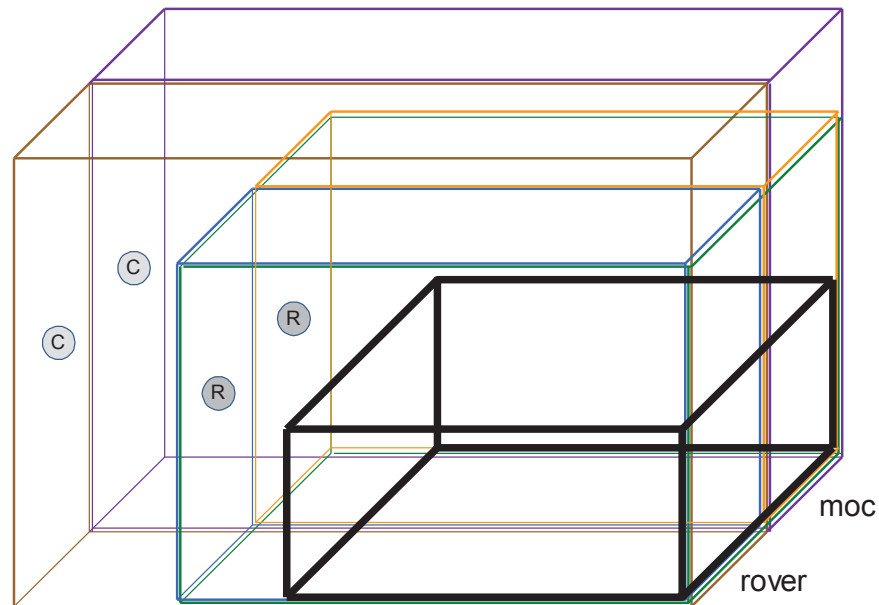


Figure 2-6: Thermal Unit Subset of Live Rover-Ops Venture

The portion of the **live rover-ops thermal** unit that resides in the **rover** continuum is a single cell as shown in figure 2-7. Modules numbered 19 and 6 are registered with the registrar of this cell.

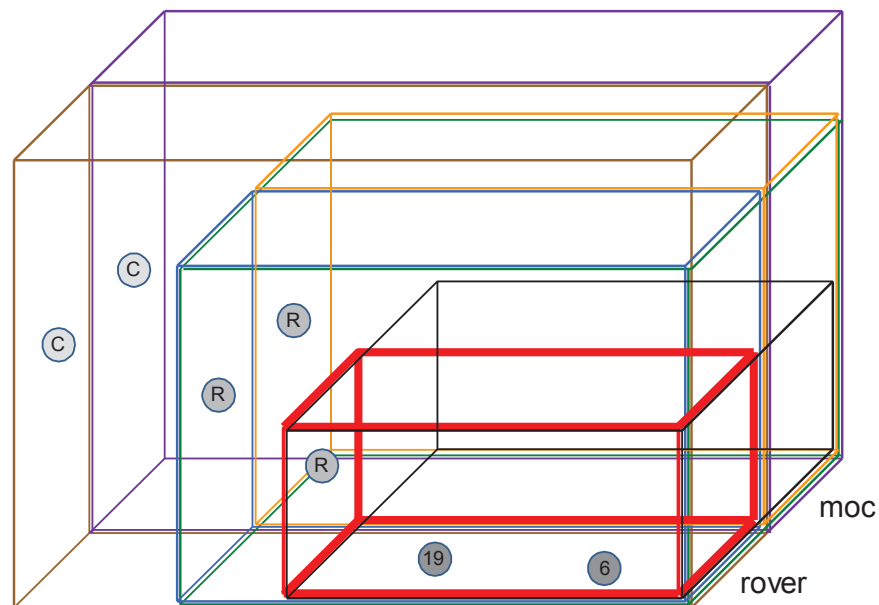


Figure 2-7: Rover Continuum Portion of Live Rover-Ops Thermal Unit

In the figure 2-8, the **live rover-ops thermal** cell residing in the **moc** continuum has been added as well.

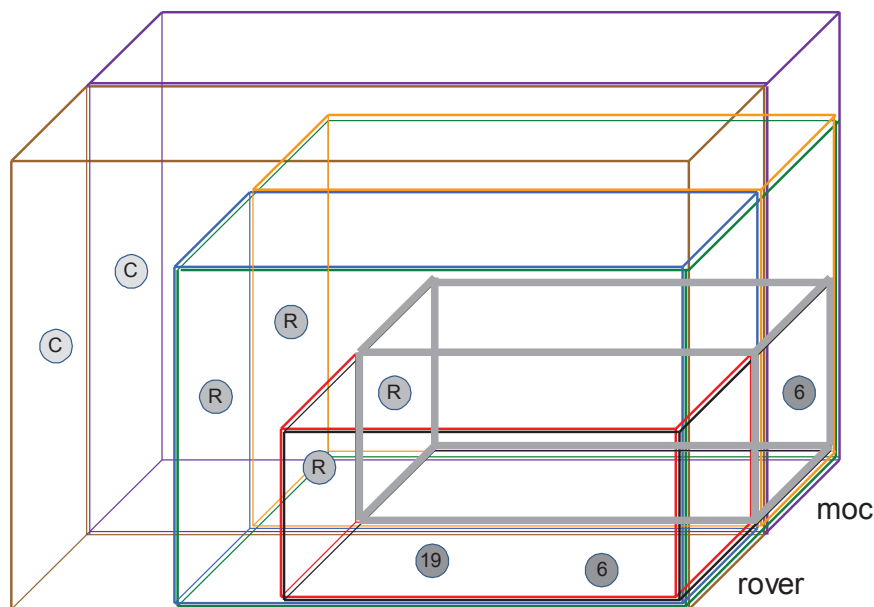


Figure 2-8: Rover and MOC Continua Portions of Live Rover-Ops Thermal Unit

It should be noted that this cell also has a registered module 6, but there is no ambiguity: module numbers are meaningful only within the scope of a cell, and the concatenation of the cell's ID (venture ID, unit ID, and continuum ID) and the module number uniquely identifies the module within the venture.

2.1.4 ARCHITECTURAL PERSPECTIVE

2.1.4.1 General

The manner in which AMS relates to other CCSDS Recommended Standards is discussed here in a continuation of the AMS communication configuration example presented in 2.1.3. In this example, AMS is to be used for monitoring and controlling the thermal subsystem (among others, potentially) of a rover on a distant planet. The design team for this mission has named the AMS application 'rover-ops' and is considering the use of AMS in either or both of two distinct computing environments, the ground data system on Earth and the rover's flight data system. AMS operations in the former environment are viewed as proceeding in a continuum named 'moc' (mission operations center), while AMS operations in the latter environment are viewed as proceeding in a continuum named 'rover'.

NOTE – Underlying protocols identified in the following subsections are examples. AMS can also run over Proximity-1 as well as any number of other protocols that meet the requirements of 3.2.

2.1.4.2 Ground Data System Example

2.1.4.2.1 Scenario

Telemetry packets acquired from the rover are passed to a module that is registered in the role of 'telemetry-sink' (module 3), which is a general-purpose publisher of telemetry information. Subject 'temperature' is used for the publication of telemetry pertaining to the thermal subsystem. Two modules in the ground data system subscribe to 'temperature': a real-time thermal state monitoring application (module 6) and a thermal state trend analysis application (module 8). Whenever a thermal telemetry packet is acquired from the rover, its contents are published by module 3 in an AAMS message on subject 'temperature' causing copies of this message to be immediately received by both module 6 and module 8.

2.1.4.2.2 Topology

Module 3 is shown here as a registered module in the 'root' cell of the message space for 'live rover-ops' in the 'moc' continuum. Modules 6 and 8 are registered in the 'thermal' cell of this message space.

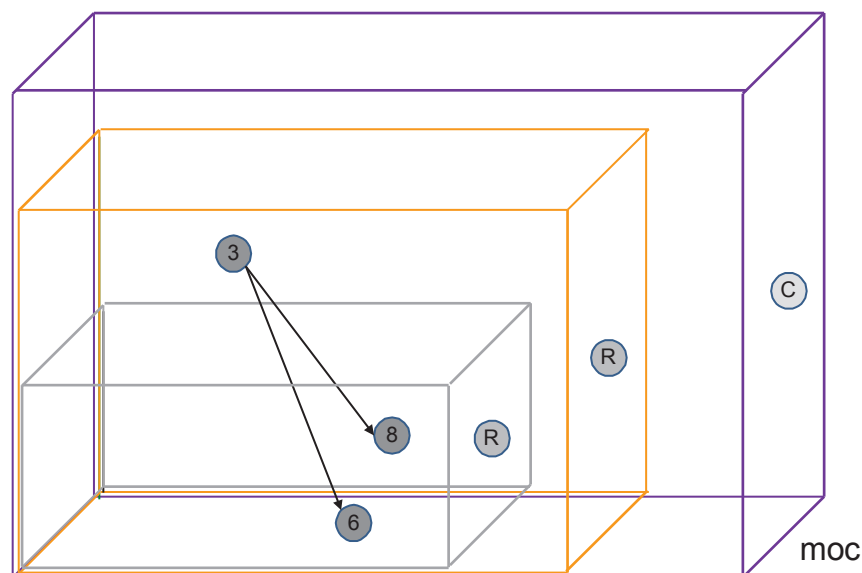


Figure 2-9: Ground Data System Registered Modules

2.1.4.2.3 Module Protocol Stack

In this example, modules 3, 6, and 8 all conform to CCSDS Spacecraft Monitor and Control architecture. The Application AMS protocol running over TCP/IP (in the mission operations center's local area network) serves as the messaging technology underlying the Messaging Abstraction Layer (MAL).

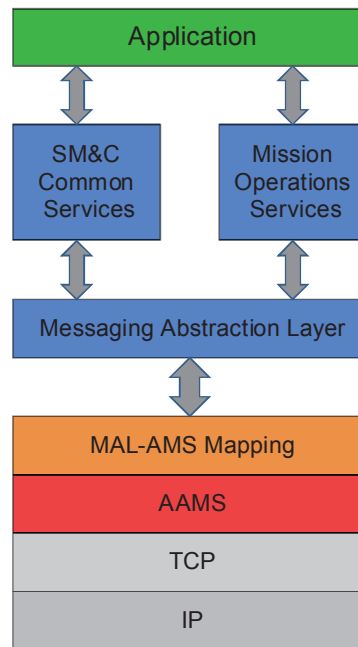


Figure 2-10: AAMS as Underlying Messaging Technology for MAL

2.1.4.3 Flight Data System Example

2.1.4.3.1 Scenario

On-board the rover, thermal sensor monitoring modules 6 and 12 periodically publish thermal sensor data in AAMS messages on subject ‘temperature’. Two on-board modules subscribe to ‘temperature’: a real-time thermal alarm module (module #19) and a module registered in the role of ‘telemetry-source’ (module #4). Whenever either module 6 or module 12 publishes a ‘temperature’ message, both module 4 and module 19 immediately receive copies. On receipt of a ‘temperature’ message, module 4 issues a thermal telemetry packet for transmission to the mission operations center on Earth.

2.1.4.3.2 Topology

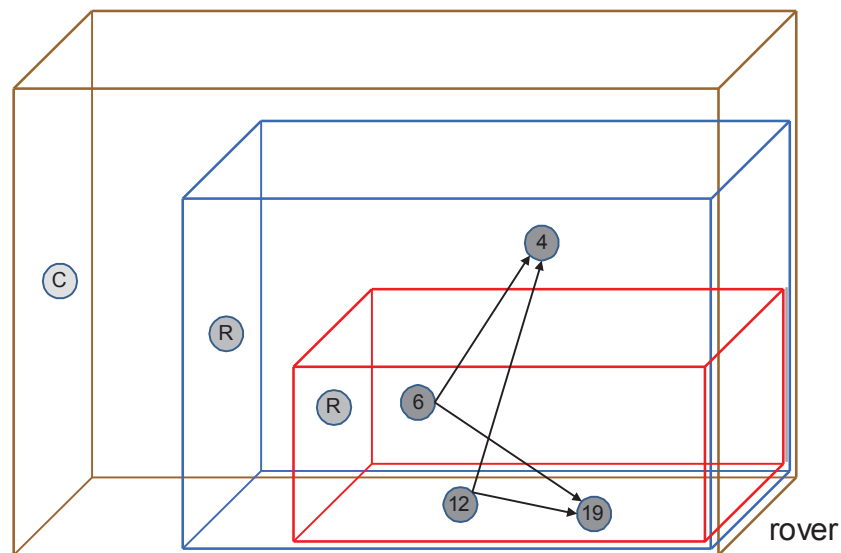


Figure 2-11: Flight Data System Registered Modules

2.1.4.3.3 Module Protocol Stack

In this example, modules 4, 6, 12, and 19 all conform to CCSDS Spacecraft Onboard Information Services (SOIS) architecture. The Application AMS protocol running over the SOIS Subnetwork Packet Service (mapped to the spacecraft’s SpaceWire bus) serves as the spacecraft’s Message Transfer Service.

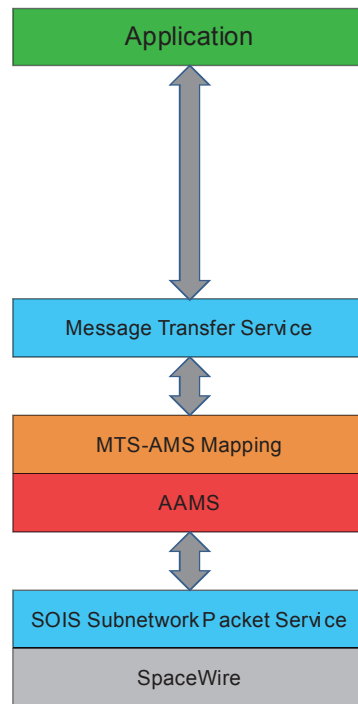


Figure 2-12: AAMS over SOIS Subnetwork Packet Service

2.1.4.4 End-to-End Mission Data System Example

2.1.4.4.1 Scenario

The mission design team have decided to use end-to-end AMS messaging to operate the rover. The mission-specific 'telemetry source' and 'telemetry sink' modules are no longer required: instead, standard RAMS gateway modules in the MOC and on the rover will manage the forwarding of command and telemetry information over the space link. Whenever either module 6 or module 12 publishes a 'temperature' message, both module 19 and module 1 (the 'live rover-ops' RAMS gateway in the 'rover' continuum) immediately receive copies. The rover's RAMS gateway uses reliable Delay/Disruption Tolerant Networking (DTN) protocols to convey the 'temperature' message to its counterpart in the MOC, which then forwards that message to the two local subscribers, modules 6 and 8.

2.1.4.4.2 Topology

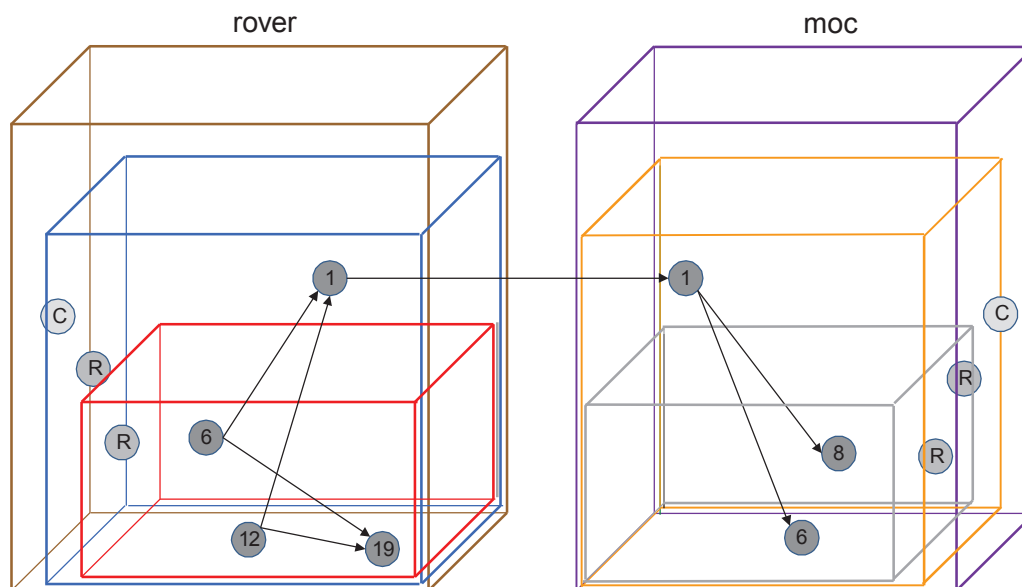


Figure 2-13: End-to-End Mission Data System Registered Modules

2.1.4.4.3 Ground Gateway Protocol Stack

In the mission operations center, the RAMS gateway uses TCP/IP for exchange of AAMS messages with other modules in its message space but uses the DTN Bundle Protocol and Licklider Transmission Protocol over CCSDS encapsulation packets and telemetry/telecommand links for exchange of RAMS messages with its counterpart on the rover.

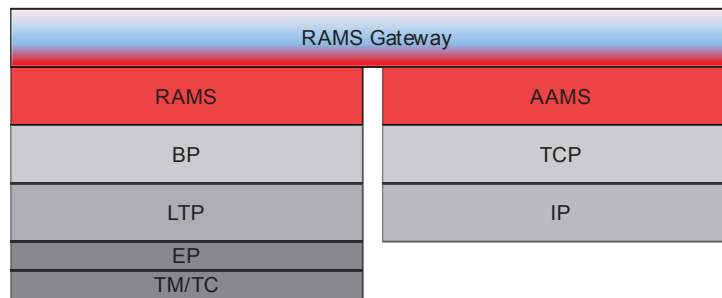


Figure 2-14: RAMS Gateway with AAMS over TCP/IP and DTN over CCSDS

2.1.4.4.4 Flight Gateway Protocol Stack

On-board the rover, the RAMS gateway uses the SOIS Subnetwork Packet Service over SpaceWire for exchange of AAMS messages with other modules in its message space but uses the DTN Bundle Protocol and Licklider Transmission Protocol over CCSDS encapsulation packets and telemetry/telecommand links for exchange of RAMS messages with its counterpart in the mission operations center.

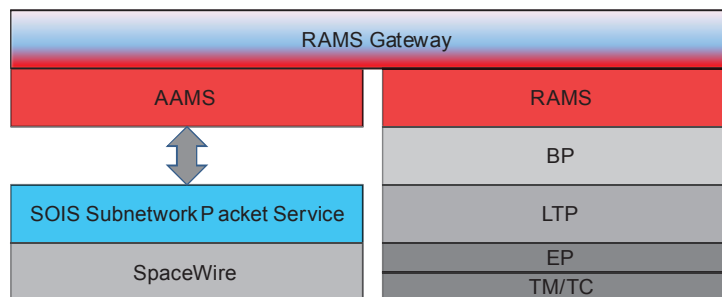


Figure 2-15: RAMS Gateway with AAMS over SOIS Subnetwork Packet Service and DTN over CCSDS

2.1.4.4.5 Enhanced Flight Module Protocol Stack

Application modules on-board the rover may use AMS either for end-to-end messaging, through Spacecraft Monitoring and Control (SM&C) services, or for local messaging.

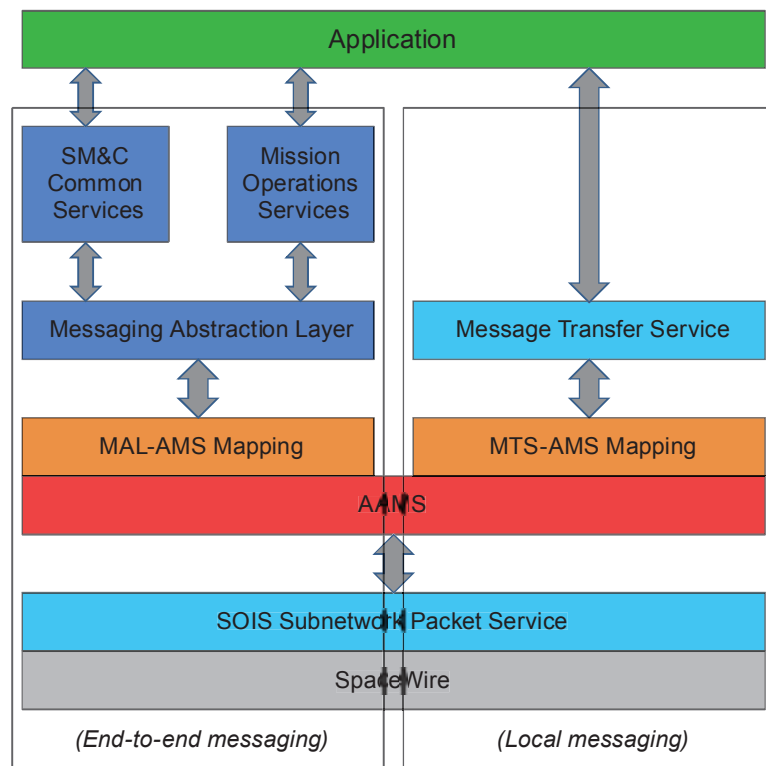


Figure 2-16: Enhanced Flight Module Protocol Stack

2.2 ARCHITECTURAL ELEMENTS

2.2.1 GENERAL

The architectural elements involved in the asynchronous message service protocols are depicted in figure 2-17 and described below.

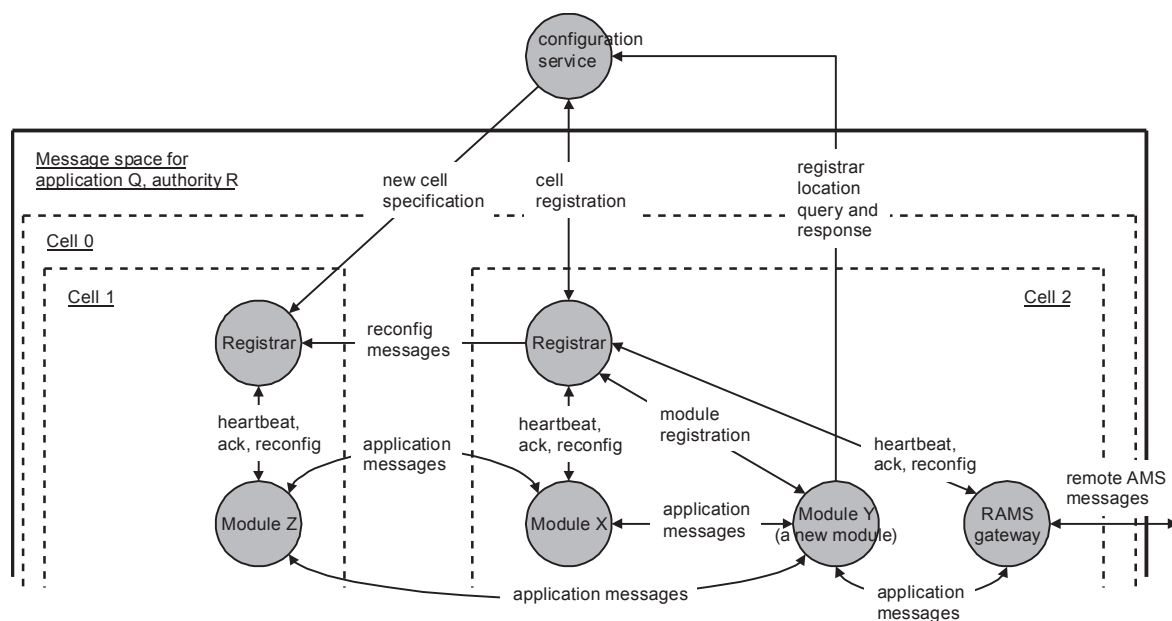


Figure 2-17: Architectural Elements of AMS (a Single Continuum)

2.2.2 COMMUNICATING ENTITIES

All AMS communication is conducted among three types of communicating entities: modules (defined earlier), registrars, and configuration servers.

A *registrar* is a communicating entity that catalogues information regarding the registered membership of a single cell of a message space. It responds to queries for this information, and it updates this information as changes are announced.

A *configuration server* is a communicating entity that manages a database of configuration information for a single continuum. In particular, it catalogues information regarding the message spaces that the continuum comprises, notably the locations of all registrars. It responds to queries for this information, and it updates this information as changes are announced.

2.3 OVERVIEW OF INTERACTIONS

2.3.1 TRANSPORT SERVICES FOR APPLICATION MESSAGES

AAMS is an OSI protocol stack layer 7 (application) service that might best be characterized as messaging ‘middleware’. As such, it relies on the capabilities of underlying Transport-layer protocols to accomplish the actual copying of a message from the memory of the sending module to the memory of the receiving module. It additionally relies on those capabilities to accomplish the transmission of the MAMS messages to and from registrars and configuration servers that enable the dynamic self-configuration of AMS message spaces.

For any given AMS continuum, some common transport service must be utilized for MAMS traffic by all communicating entities involved in the operations of all message spaces in the continuum. The transport service selected for this purpose is termed the continuum’s *Primary Transport Service (PTS)*.

The PTS clearly can also be used for application message exchange among all modules in a continuum, as it must be universally available for MAMS message exchange. In some cases, however, improved application performance can be achieved by using a different transport service for message exchange between modules that share access to some especially convenient communication medium, such as a shared-memory message queue. These performance-optimizing transport services are termed *Supplementary Transport Services (STSeS)*.

The service access point at which a module can receive messages via a given transport service is termed the module’s *delivery point* for that service. Multiple delivery points (for different transport services) may be characterized by the same *service mode*, that is, by the same levels of transmission order preservation and diligence in message delivery. For a given service mode, the list of all delivery points that provide that mode of service to a given module—in descending order of preference (generally, descending order of network performance)—is termed the *delivery vector* for that service mode, for that module.

2.3.2 REGISTRAR REGISTRATION

Every message space always comprises at least one cell (the root cell), and each module resides within (is registered in) some cell; in the simplest case all modules of the message space reside in the root cell. Each cell is served by a single registrar, which is responsible for monitoring the health of all modules in the cell’s registered membership and for propagating six kinds of message space configuration changes: module registrations and terminations, subscription and invitation assertions, and subscription and invitation cancellations. On reception of one of these reconfiguration messages from a module in its own cell’s registered membership, the registrar immediately propagates the message to the other modules in that registered membership and then to the registrars of other cells in the message space, as appropriate; on receiving such a message from a remote cell’s registrar, the registrar propagates it to the modules in its own cell’s registered membership.

The registrars themselves register with the configuration server for the continuum within which the message space is contained. A list of all possible network locations for the configuration server, ordered by some mechanism that reflects the continuum's globally recognized strict relative order among candidate configuration servers, must be 'well known'—that is, included in the AMS MIBs exposed to all registrars for all message spaces in the continuum—and each continuum must have a configuration server in operation at one of those locations at all times in order to enable registrars and modules to register. (The manner in which this latter requirement is satisfied is an implementation matter.)

All registrars and modules of the same message space must register through the same configuration server. The registrars and modules for any number of different message spaces may register with the same configuration server.

2.3.3 MODULE REGISTRATION

Each module has a single associated *Meta-AMS Delivery Point (MADP)* at which the module is prepared to receive MAMS messages. A new module joins a message space by registering itself within some cell of the message space, i.e., by announcing its role name and its MADP to the cell's registrar and thereby inserting itself into the cell's registered membership. However, knowledge of how to communicate with that registrar cannot be hard-coded into the module because the relevant registrar might be running at different network locations at different times.

For this reason, the first step in registering a new module is to contact the configuration server at one of its well-known possible network locations; as with the registrars, a list of all possible network locations for the configuration server, in order of descending preference, must be included in the AMS MIBs exposed to all application modules. The configuration server tells the new module how to contact its registrar. The module obtains a *module number* (which uniquely identifies it within the cell's registered membership) from the registrar and thereby registers. The registrar ensures that all other modules in the message space learn the new module's role name, module number, and MADP. Declaration of those modules' own role names, module numbers, and MADPs to the new module is then accomplished in one of two ways, depending on implementation: either the modules themselves send this information to the new module or else the registrar retrieves this information from its local cache and sends it directly to the new module.

2.3.4 MONITORING MODULE HEALTH

In order to acquire and maintain accurate knowledge of the configuration of a message space (for application purposes, and also to avoid wasting resources on attempts to send messages to nonexistent modules or per concluded subscriptions), it is important for each registrar always to detect the terminations of modules in its cell's registered membership. When a module terminates under application control it automatically notifies its registrar that it is stopping. However, if a module crashes, or the host on which a module resides is simply powered off or rebooted, no such notification is transmitted to the registrar. For this reason, every module automatically sends a 'heartbeat' message to its registrar every N4 seconds. (See table 1-1 for definitions of N3, N4, and N6.) The registrar interprets N6 successive missing heartbeats as an indication that the module has terminated.

Whenever it detects the termination of a module (either an overt termination or a termination imputed from heartbeat failure), the registrar informs all other modules in the cell's registered membership and, via other registrars, all other modules in the message space of the module's demise.

When the termination is imputed from a heartbeat failure, the registrar also tries to send a message to the terminated module telling it that it has been presumed dead; if this module is in fact still running (perhaps it had merely hung trying to write on a blocked file descriptor), it terminates immediately on reception of this message. This minimizes system confusion resulting from other application behavior that may have been triggered by the imputed termination.

NOTE – The value of N4 is defined as twice the value of N3, which is a global implementation parameter within any single continuum. If a very large value of N3 is selected, then heartbeat exchange is in effect disabled; that is, the monitoring of module health, registrar health, and configuration server health is optional. AMS entity health monitoring may be unnecessary under certain circumstances, e.g., in tightly controlled spacecraft on-board environments where other mechanisms for the detection of module failure are available. In these environments, the initiation of AMS procedures for handling imputed entity termination—in effect, the simulation of heartbeat failure—is an implementation matter.

2.3.5 MONITORING REGISTRAR HEALTH

In addition to monitoring the heartbeats of all modules in its cell's registered membership, each registrar issues its own heartbeats to those modules on the same cycle. Each module interprets N6 successive missing registrar heartbeats as an indication that the registrar itself has crashed. On detecting a registrar crash, the module presumes that the registrar has been restarted since it crashed; it re-queries the configuration server to determine the new network location of the registrar, reconnects to the registrar, and resumes exchanging heartbeats.

This presumption is reasonable because the reciprocal heartbeat monitoring relationship between a registrar and its modules is replicated in the relationship between the configuration server and all registrars, but on a slightly shorter cycle. The configuration server interprets $N6$ successive missing registrar heartbeats as an indication that the registrar has crashed; on detecting such a crash it can automatically take some implementation-specific action to cause the registrar to be restarted, possibly on a different host, so by the time the registrar's modules detect its demise it should already be running again.

Since the module heartbeat interval is $N4$ seconds, within the first $N5$ seconds after restart the registrar will have received heartbeat messages from every module in the cell's registered membership that is still running and will therefore know accurately the configuration of the cell.

NOTE – The worst-case interval between failure and recovery of registration service within a cell is $N5$ seconds. In the event that $N3$ is a very large number (disabling heartbeat exchange, as discussed above), implementation-specific mechanisms for simulating the effects of heartbeat failure will reduce this recovery interval to far less than $N5$ seconds.

This accurate configuration information must be delivered to new modules at the time they start up (so that they in turn are qualified to orient a newly restarted registrar to the cell's configuration in the event that the registrar crashes). For this reason, during the first $N5$ seconds (or other implementation-specific recovery interval, as noted above) after the registrar starts it accepts only MAMS messages from modules that are already registered in the cell (i.e., have been assigned module numbers); if it accepted and processed a registration message from a new module before being certain of the status of all old ones, it would run the risk of delivering incorrect information to the new module.

2.3.6 CONFIGURATION SERVICE FAIL-OVER

It is of course also possible for a configuration server to be killed (or for its host to be rebooted, etc.). Each registrar interprets $N6$ successive missing configuration server heartbeats as an indication that the configuration server has crashed. On detecting such a crash, the registrar begins cycling through all of the well-known possible network locations for the continuum's configuration server, trying to re-establish communication after the server's restart, possibly at an alternate network location. Moreover, so long as the configuration server is down no new module can join any message space in the continuum, since new modules will have no way of knowing the network locations of their intended registrars; they too will be cycling through all of the well-known possible network locations for the continuum's configuration server, trying to establish initial communications.

When the configuration server is restarted at one of its well-known possible network locations, however, all registrars will eventually¹ find it and re-announce themselves to it; new module registration activity will thereupon resume.

It is possible, in this sort of failure scenario, that multiple configuration servers may be operating concurrently for a brief time; for example, the perceived failure of a configuration server might have been caused by a transient network connectivity failure rather than an actual server crash. To resolve this sort of situation, each running configuration server periodically sends an 'I am running' MAMS message to every lower-ranking configuration server network location in the well-known list of configuration server locations. When a configuration server receives such a message, it immediately terminates; all registrars and modules that were communicating with it will detect its disappearance and search again for the highest-ranking reachable configuration server, eventually bringing the continuum back to orderly operations.

2.3.7 CONFIGURATION RESYNCHRONIZATION

Finally, every registrar can optionally be configured to re-advertise to the entire message space the detailed configuration of its cell's registered membership (all active modules, all subscriptions and invitations) at some user-specified frequency, e.g., once per minute. This capability is referred to as *configuration resynchronization*. Configuration resynchronization of course generates additional message traffic, and it may be unnecessary in extremely simple or extremely stable operating environments. But it does ensure that every change in message space configuration will eventually be propagated to every module in the message space, even if some MAMS messages are lost and even if an arbitrary number of registrars had crashed at the time the change occurred.

Taken together, these measures make AMS applications tolerant of several types of faults:

- When a module crashes, its registrar detects the loss of heartbeat within N6 heartbeat intervals and notifies the rest of the message space. Application message transmission among the remaining modules in the message space is unaffected.
- When a registrar crashes, its configuration server detects the loss of heartbeat within N6 heartbeat intervals and takes action to restart the registrar. During the time that the cell has no registrar, transmission of application messages among modules of the message space is unaffected, but the heartbeat failures of crashed modules are not detected and reconfiguration messages issued by the modules in the cell (registrations, terminations, subscription and invitation assertions, and subscription and invitation cancellations) are not propagated to any other modules. However, after the registrar is restarted it will eventually detect the losses of heartbeat from all crashed modules and will issue obituaries to the message space; in addition, if configuration

¹ The AMS Green Book contains notes on an algorithm for estimating the worst-case elapsed time prior to re-establishment of module registration activity.

resynchronization is enabled it will eventually re-propagate the lost reconfiguration messages.

- When a configuration server crashes, registration of new modules ceases until the configuration server is restarted. But no application modules fail (at least, not because of communication failure), and on restart of the configuration server all new module registration eventually resumes.

2.3.8 SECURITY

NOTE – *The Application of CCSDS Protocols to Secure Systems* (CCSDS 350.0-G-2) should be consulted for definitions of the terms used in this subsection.

Optionally, AMS can be configured to confine service access to application modules that can prove they are authorized to participate. For this purpose, asymmetric MAMS encryption may be used as follows:

- The AMS MIB exposed to the configuration server contains a list of all applications for which registration service may be offered, identified by application name. Associated with each application name is the AMS public encryption key for that application.
- The AMS MIB exposed to every registrar in each message space contains a list of all functional role names defined for the message space's application; this list limits the role names under which modules may register in that message space. Associated with each role name is the AMS public encryption key for the application module(s) that may register in that role.
- The AMS MIBs exposed to all registrars and application modules in the message space contain the AMS public encryption key of the configuration server.
- The AMS MIBs exposed to the configuration server and to all registrars and application modules in the message space contain the private encryption keys that are relevant to those entities.

As described later, this information is used to authenticate registrar registration and exclude spurious registrars from the message space, to authenticate module registration attempts and deny registration to unauthorized application modules, and to assure the authenticity, confidentiality, and integrity of MAMS traffic exchanged between modules and their registrars.

In addition, the confidentiality and integrity of AMS message exchange may be protected at subject granularity. The AMS MIB exposed to each module of a given message space may contain, for any subset of the message subjects (identified by name and number) used in the message space's application: