



IEEE Standard for Spectrum Sensing Interfaces and Data Structures for Dynamic Spectrum Access and Other Advanced Radio Communication Systems

Amendment 2: Spectrum Database Interfaces

IEEE Communications Society

Developed by the
Dynamic Spectrum Access Networks Standards Committee

IEEE Std 1900.6b™-2022
(Amendment to IEEE Std 1900.6™-2011)

IEEE Standard for Spectrum Sensing Interfaces and Data Structures for Dynamic Spectrum Access and Other Advanced Radio Communication Systems

Amendment 2: Spectrum Database Interfaces

Developed by the

Dynamic Spectrum Access Networks Standards Committee
of the
IEEE Communications Society

Approved 16 June 2022

IEEE SA Standards Board

Abstract: This amendment to IEEE Std 1900.6-2011 adds procedures, protocols, and message format specifications for the exchange of sensing related data, control data, and configuration data between spectrum sensors and spectrum databases. In addition, it adds specifications for the exchange of sensing related data between the data archive and spectrum databases, and the cognitive engine and spectrum databases.

Keywords: amendment, IEEE 1900.6™, IEEE 1900.6b™, interface, spectrum sensing, spectrum database

The Institute of Electrical and Electronics Engineers, Inc.
3 Park Avenue, New York, NY 10016-5997, USA

Copyright © 2022 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 30 September 2022. Printed in the United States of America.

IEEE is a registered trademark in the U.S. Patent & Trademark Office, owned by The Institute of Electrical and Electronics Engineers, Incorporated.

PDF: ISBN 978-1-5044-8844-0 STD25520
Print: ISBN 978-1-5044-8845-7 STDPD25520

IEEE prohibits discrimination, harassment, and bullying. For more information, visit <http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html>.

No part of this publication may be reproduced in any form, in an electronic retrieval system or otherwise, without the prior written permission of the publisher.

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE Standards documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page (<https://standards.ieee.org/ipr/disclaimers.html>), appear in all standards and may be found under the heading “Important Notices and Disclaimers Concerning IEEE Standards Documents.”

Notice and Disclaimer of Liability Concerning the Use of IEEE Standards Documents

IEEE Standards documents are developed within IEEE Societies and subcommittees of IEEE Standards Association (IEEE SA) Board of Governors. IEEE develops its standards through an accredited consensus development process, which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEE Standards are documents developed by volunteers with scientific, academic, and industry-based expertise in technical working groups. Volunteers are not necessarily members of IEEE or IEEE SA and participate without compensation from IEEE. While IEEE administers the process and establishes rules to promote fairness in the consensus development process, IEEE does not independently evaluate, test, or verify the accuracy of any of the information or the soundness of any judgments contained in its standards.

IEEE makes no warranties or representations concerning its standards, and expressly disclaims all warranties, express or implied, concerning this standard, including but not limited to the warranties of merchantability, fitness for a particular purpose and non-infringement. In addition, IEEE does not warrant or represent that the use of the material contained in its standards is free from patent infringement. IEEE standards documents are supplied “AS IS” and “WITH ALL FAULTS.”

Use of an IEEE standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

In publishing and making its standards available, IEEE is not suggesting or rendering professional or other services for, or on behalf of, any person or entity, nor is IEEE undertaking to perform any duty owed by any other person or entity to another. Any person utilizing any IEEE Standards document, should rely upon his or her own independent judgment in the exercise of reasonable care in any given circumstances or, as appropriate, seek the advice of a competent professional in determining the appropriateness of a given IEEE standard.

IN NO EVENT SHALL IEEE BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO: THE NEED TO PROCURE SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE PUBLICATION, USE OF, OR RELIANCE UPON ANY STANDARD, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE AND REGARDLESS OF WHETHER SUCH DAMAGE WAS FORESEEABLE.

Translations

The IEEE consensus development process involves the review of documents in English only. In the event that an IEEE standard is translated, only the English version published by IEEE is the approved IEEE standard.

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, nor be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that the presenter's views should be considered the personal views of that individual rather than the formal position of IEEE, IEEE SA, the Standards Committee, or the Working Group. Statements made by volunteers may not represent the formal position of their employer(s) or affiliation(s).

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE or IEEE SA. However, **IEEE does not provide interpretations, consulting information, or advice pertaining to IEEE Standards documents.**

Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its Societies and subcommittees of the IEEE SA Board of Governors are not able to provide an instant response to comments, or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in evaluating comments or in revisions to an IEEE standard is welcome to join the relevant IEEE working group. You can indicate interest in a working group using the Interests tab in the Manage Profile & Interests area of the [IEEE SA myProject system](#).¹ An IEEE Account is needed to access the application.

Comments on standards should be submitted using the [Contact Us](#) form.²

Laws and regulations

Users of IEEE Standards documents should consult all applicable laws and regulations. Compliance with the provisions of any IEEE Standards document does not constitute compliance to any applicable regulatory requirements. Implementers of the standard are responsible for observing or referring to the applicable regulatory requirements. IEEE does not, by the publication of its standards, intend to urge action that is not in compliance with applicable laws, and these documents may not be construed as doing so.

Data privacy

Users of IEEE Standards documents should evaluate the standards for considerations of data privacy and data ownership in the context of assessing and using the standards in compliance with applicable laws and regulations.

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under US and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption

¹ Available at: <https://development.standards.ieee.org/myproject-web/public/view.html#landing>.

² Available at: <https://standards.ieee.org/content/ieee-standards/en/about/contact/index.html>.

by public authorities and private users, neither IEEE nor its licensors waive any rights in copyright to the documents.

Photocopies

Subject to payment of the appropriate licensing fees, IEEE will grant users a limited, non-exclusive license to photocopy portions of any individual standard for company or organizational internal use or individual, non-commercial use only. To arrange for payment of licensing fees, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400; <https://www.copyright.com/>. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

Updating of IEEE Standards documents

Users of IEEE Standards documents should be aware that these documents may be superseded at any time by the issuance of new editions or may be amended from time to time through the issuance of amendments, corrigenda, or errata. An official IEEE document at any point in time consists of the current edition of the document together with any amendments, corrigenda, or errata then in effect.

Every IEEE standard is subjected to review at least every 10 years. When a document is more than 10 years old and has not undergone a revision process, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE standard.

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit [IEEE Xplore](#) or [contact IEEE](#).³ For more information about the IEEE SA or IEEE's standards development process, visit the IEEE SA Website.

Errata

Errata, if any, for all IEEE standards can be accessed on the [IEEE SA Website](#).⁴ Search for standard number and year of approval to access the web page of the published standard. Errata links are located under the Additional Resources Details section. Errata are also available in [IEEE Xplore](#). Users are encouraged to periodically check for errata.

Patents

IEEE Standards are developed in compliance with the [IEEE SA Patent Policy](#).⁵

Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken by the IEEE with respect to the existence or validity of any patent rights in connection therewith. If a patent holder or patent applicant has filed a statement of assurance via an Accepted Letter of Assurance, then the statement is listed on the IEEE SA Website at <https://standards.ieee.org/about/sasb/patcom/patents.html>. Letters of Assurance may indicate whether the Submitter is willing or unwilling to grant licenses under patent rights without compensation or under reasonable rates, with reasonable terms and conditions that are demonstrably free of any unfair discrimination to applicants desiring to obtain such licenses.

³ Available at: <https://ieeexplore.ieee.org/browse/standards/collection/ieee>.

⁴ Available at: <https://standards.ieee.org/standard/index.html>.

⁵ Available at: <https://standards.ieee.org/about/sasb/patcom/materials.html>.

Essential Patent Claims may exist for which a Letter of Assurance has not been received. The IEEE is not responsible for identifying Essential Patent Claims for which a license may be required, for conducting inquiries into the legal validity or scope of Patents Claims, or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance, if any, or in any licensing agreements are reasonable or non-discriminatory. Users of this standard are expressly advised that determination of the validity of any patent rights, and the risk of infringement of such rights, is entirely their own responsibility. Further information may be obtained from the IEEE Standards Association.

IMPORTANT NOTICE

IEEE Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. IEEE Standards development activities consider research and information presented to the standards development group in developing any safety recommendations. Other information about safety practices, changes in technology or technology implementation, or impact by peripheral systems also may be pertinent to safety considerations during implementation of the standard. Implementers and users of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

Participants

At the time this standard was submitted to the IEEE SA Standards Board for approval, the IEEE 1900.6 Working Group had the following membership:

Oliver Holland, *Chair*
Luzango Mfupe, *Vice-Chair*
Konstantinos Katzis, *Secretary*
Guoru Ding, Konstantinos Katzis, Oliver Holland, *Technical Editors*

The following members of the individual balloting committee voted on this standard. Balloters may have voted for approval, disapproval, or abstention.

Sandeep Agrawal	SangKwon Jeong	Brian Petry
Khalid Al-Mufti	Stuart Kerry	Walter Pieniac
Francesco Benedetto	Yongbum Kim	Clinton Powell
Vern Brethour	Shoichi Kitazawa	Venkatesha Prasad
Demetrio Bucaneg, Jr	Mieczyslaw Kokar	R. K. Rannow
C. Caicedo Bastidas	Jan Kruys	Ranga Reddy
Paul Cardinal	Hyeong Ho Lee	Tony Rennie
Juan Carreon	Kin Hong Lee	Benjamin Rolfe
Pin Chang	Arthur H. Light	Naotaka Sato
Marc Emmelmann	Javier Luiso	Reinhard Schrage
Sho Furuichi	Michael Lynch	Dorothy Stanley
Randall Groves	Richard Maiden	John Vergis
Michael Gundlach	Brett McClellan	Lisa Ward
Marco Hernandez	Michael Montemurro	Scott Willy
Lili Hervieu	Biodun Osunkunle	Andreas Wolf
Werner Hoelzl	Bansi Patel	Yu Yuan
Oliver Holland	Arumugam Paventhan	Oren Yuen
Atsushi Ito		Andrew Zambell

When the IEEE SA Standards Board approved this standard on 16 June 2022, it had the following membership:

David J. Law, *Chair*
Ted Burse, *Vice Chair*
Gary Hoffman, *Past Chair*
Konstantinos Karachalios, *Secretary*

Edward A. Addy	Johnny Daozhuang Lin	Mark Siira
Ramy Ahmed Fathy	Kevin Lu	Dorothy V. Stanley
J. Travis Griffith	Daleep C. Mohla	Lei Wang
Guido R. Hiertz	Andrew Myles	F. Keith Waters
Yousef Kimiagar	Damir Novosel	Karl Weber
Joseph L. Koepfinger*	Annette D. Reilly	Sha Wei
Thomas Koshy	Robby Robson	Philip B. Winston
John D. Kulick	Jon Walter Rosdahl	Daidi Zhong

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 1900.6b-2002, IEEE Standard for Spectrum Sensing Interfaces and Data Structures for Dynamic Spectrum Access and Other Advanced Radio Communication Systems—Amendment 2: Spectrum Database Interfaces.

This amendment to IEEE Std 1900.6 adds procedures, protocols, and message format specifications for the exchange of sensing related data, control data and configuration data between spectrum sensors and spectrum databases. In addition, it provides recommendations for the exchange of sensing related data between the data archive and spectrum databases, and the cognitive engine and spectrum databases.

Contents

1. Overview	7
1.4 Word usage	7
2. Normative references	8
3. Definitions, acronyms, and abbreviations	8
3.1 Definitions	8
3.2 Acronyms and abbreviations	9
4. System model	9
4.5 Scenario 5: Single CE/DA, single Sensor and a single CE, DA, or Sensor hosting a gateway function as a service interface towards a client spectrum database	9
4.6 Scenario 6: Single CE/DA, single Sensor and spectrum database as an IEEE 1900.6 compliant entity (SD)	11
5. The IEEE 1900.6 reference model	13
5.1 General description	13
5.1a Security requirements	15
5.1b Entity identification method	16
5.3 Service access points	17
6. Information description	80
6.3 Description of sensing-related parameters	81
6.3a Description of SAP exception and error codes	83
7. State diagram and generic procedures	86
7.5 Service access point life cycle	86
Annex A (informative) Use cases	90
A.2 Use cases from the perspective of spectrum sensing	90
A.3 Use cases involving spectrum databases	90
Annex B (informative) Use case classification	112
Annex DA (informative) IEEE 1900.6 SD: Scope and usage	119
DA.1 Introduction	119
DA.2 Scope and usage of the SD	119
DA.3 SD requirements	120
DA.4 SD interfaces	122
DA.5 SD deployment examples	124
DA.6 Best practice referencing and addressing IEEE 1900.6 entities	132
Annex DB (informative) Entity list use and management best practice	135
Annex DC (informative) Security considerations	139
Annex DD (informative) IEEE 1900.6 SAP Information model and best-practice architecture	143
DD.1 Adaptation layer	143
DD.2 SAP Management	144
DD.3 SAP Logic	144
DD.4 Security sublayer	144
DD.5 Communication system abstraction layer	144
DD.6 Simplifications for trusted subsystems	145
DD.7 Sophisticated and special-use SAP adaptation layers	146
Annex F (informative) Bibliography	147

IEEE Standard for Spectrum Sensing Interfaces and Data Structures for Dynamic Spectrum Access and Other Advanced Radio Communication Systems

Amendment 2: Spectrum Database Interfaces

NOTE—The editing instructions contained in this **amendment** define how to merge the material contained therein into the existing base standard and its amendments to form the comprehensive standard.

The editing instructions are shown in ***bold italic***. Four editing instructions are used: change, delete, insert, and replace. ***Change*** is used to make corrections in existing text or tables. The editing instruction specifies the location of the change and describes what is being changed by using ~~strike through~~ (to remove old material) and underscore (to add new material). ***Delete*** removes existing material. ***Insert*** adds new material without disturbing the existing material. Insertions may require renumbering. If so, renumbering instructions are given in the editing instruction. ***Replace*** is used to make changes in figures or equations by removing the existing figure or equation and replacing it with a new one. Editing instructions, change markings, and this NOTE will not be carried over into future editions because the changes will be incorporated into the base standard.

1. Overview

Delete the existing 1.4 and replace it with the following new subclause 1.4:

1.4 Word usage

The word *shall* indicates mandatory requirements strictly to be followed in order to conform to the standard and from which no deviation is permitted (*shall* equals *is required to*).^{1,2}

The word *should* indicates that among several possibilities one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required (*should* equals *is recommended that*).

¹ The use of the word *must* is deprecated and might not be used when stating mandatory requirements, *must* is used only to describe unavoidable situations.

² The use of *will* is deprecated and might not be used when stating mandatory requirements, *will* is only used in statements of fact.

The word *may* is used to indicate a course of action permissible within the limits of the standard (*may equals is permitted to*).

The word *can* is used for statements of possibility and capability, whether material, physical, or causal (*can equals is able to*).

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used, so each referenced document is cited in text and its relationship to this document is explained). For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments or corrigenda) applies.

Insert the following new reference in Clause 2 and reorder alphanumerically:

IETF RFC 4122 (July 2005), A Universally Unique Identifier (UUID) URN Namespace, Leach, P., Mealling, M., and Salz, R.³

3. Definitions, acronyms, and abbreviations

For the purposes of this document, the following terms and definitions apply. The *IEEE Standards Dictionary Online* should be consulted for terms not defined in this clause.⁴

3.1 Definitions

Insert the following new definition in alphabetical order:

spectrum database: A function providing information on the spectrum space under consideration.

NOTE 1—A spectrum database might consist of one of more of the following characteristics: a structured set of spectrum-related data, a set of algorithms operating on that data, a number of interfaces to information sources providing that data, and a number of interfaces to clients using that data, perhaps in a processed form.

NOTE 2—This might be embodied in a permanent form or instantiated on demand in the form of dedicated functions potentially distributed over infrastructure.

NOTE 3—Such a database is typically used for storage, processing, and management of spectrum related data.

NOTE 4—Spectrum-related data might be provided by modeling, planning, or sensing and has the capability of being used for protection of spectrum users against potential harm through interference by other spectrum users or to optimize spectrum utilization.

NOTE 5—The spectrum database might communicate with legacy or non-legacy sensors. It might also communicate with gateways and other databases and might act as a mediator supporting spectrum sensing information exchange, applicable to distributed spectrum sensing scenarios. Spectrum databases are expected to be centrally managed by a private or public entity that is accessible in various ways from other sensors through gateways, cognitive engines, or even other spectrum databases.

³ IETF RFCs are available from the Internet Engineering Task Force (<http://www.ietf.org/rfc.html>).

⁴ IEEE Standards Dictionary Online subscription is available at: <http://dictionary.ieee.org>.

3.2 Acronyms and abbreviations

Insert the following abbreviation in alphabetical order:

CBRS	Citizens Broadband Radio Service
DoS	denial of service
EIRP	Equivalent Isotropically Radiated Power
ESC	Environmental Sensing Capability
HAPS	High Altitude Platforms
ICV	integrity check value
IE	incumbent emulation
LSA	Licensed-Shared Access
MIC	Message Integrity Checking
MITM	Man-in-the-Middle attack
MVNO	mobile virtual network operator
SAS	Spectrum Access System
SD	spectrum database
SLAAC	stateless address auto configuration
SSDF	spectrum sensing data falsification
TKIP	Temporal Key Integrity Protocol
TVWS	TV white space
WSD	white space device

4. System model

Insert the following new subclauses after 4.4:

4.5 Scenario 5: Single CE/DA, single Sensor and a single CE, DA, or Sensor hosting a gateway function as a service interface towards a client spectrum database

In this scenario, shown in Figure 3b, an IEEE 1900.6 compliant entity is hosting a gateway function that publishes an interface towards a spectrum database aiming to utilize an IEEE 1900.6 compliant distributed sensing system.

NOTE 1—In general, any IEEE 1900.6 entity can host the gateway function, but certain IEEE 1900.6 entities might be better suited than others depending on the particular database application.

An IEEE 1900.6 distributed sensing system aiming to assist a spectrum database shall provide at least one of the following options:

- An IEEE 1900.6 compliant Sensor entity hosting a spectrum database gateway function that shall provide interfaces and functions to convey sensing control information and to obtain sensing related information from one or more local or remote Sensors.
- An IEEE 1900.6 compliant CE entity hosting a spectrum database gateway function that shall provide interfaces and functions to convey sensing control information and to obtain sensing related information from one or more local or remote Sensors. It shall provide an implementation-dependent number of CE specific functions that may process sensing related information from multiple Sensors or may provide assistance in coordinating multiple Sensors in use cases where collaborative sensing is required.
- An IEEE 1900.6 compliant DA entity hosting a spectrum database gateway function that shall provide interfaces and functions to convey sensing control information and to obtain sensing related information from one or more local or remote Sensors. It shall provide an implementation-dependent number of DA specific functions that may allow a spectrum database to provide sensing control information, in particular regulatory information and policies to a distributed sensing system.

NOTE 2—This standard does not exclude any implementation that merges two or more of the above functions into a single instance, or that implements any of them in parallel to obtain the desired functionality for a certain application, or that provides gateway functions towards more than one spectrum database.

The gateway function towards a spectrum database is considered implementation-dependent and is out-of-scope of this standard. This is also the case for any interface between the gateway function and spectrum database to be provided by the gateway function.

NOTE 3—Some best-practice sample approaches for such interfaces are given in Annex DA.

Since the gateway function hides the existence of the non-compliant subsystem (i.e., the Spectrum Database Application), and the hosting IEEE 1900.6 entity (i.e., the Sensor, CE, or DA) is compliant with this standard. This scenario is a “1:N scenario” as given by 4.2 (i.e., the spectrum database application is hosted by the only CE/DA, or it is hosted by a Sensor while there is another CE/DA present), or it is an “M:N scenario” as given by 4.3 with respect to the IEEE 1900.6 interfaces.

The following instances of the IEEE 1900.6 logical interfaces are involved in this scenario:

- CE/DA-S interface
- CE-CE/DA interface
- S-S interface

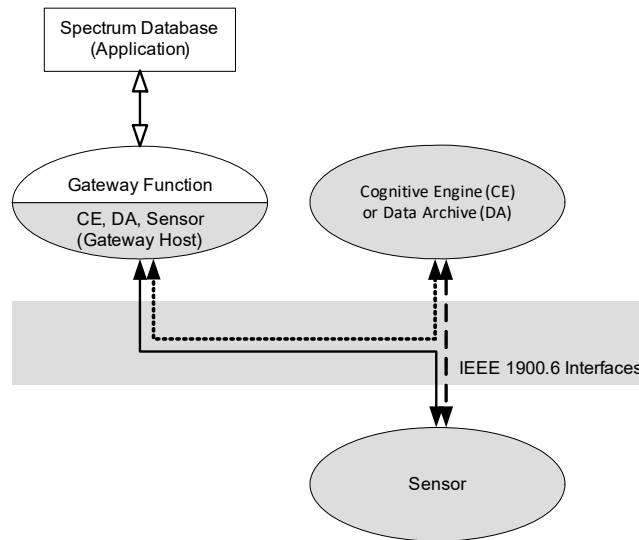


Figure 3b—Single IEEE 1900.6 entity (Sensor, CE, or DA) hosting a proprietary gateway function to provide an interface towards a spectrum database

4.6 Scenario 6: Single CE/DA, single Sensor and spectrum database as an IEEE 1900.6 compliant entity (SD)

In the scenario depicted in Figure 3c, a spectrum database is acting as an IEEE 1900.6 compliant entity implementing a proprietary spectrum database application that uses IEEE 1900.6 service access points. The service access points shall be provided by the hosting entity as defined in Clause 5.

NOTE 1—This scenario might be given preference over scenario 5 in cases where requirements exist that demand a tight integration of spectrum database and distributed sensing system, such as a high trust level of the integrated system or a certification required for the integrated system. For scenario 6, no exposed interfaces exist, and exclusive resource access can be achieved by means internal to the integrated system.

An IEEE 1900.6 compliant Spectrum Database (SD) entity shall provide one of the following options to integrate a spectrum database application (cf. 4.5) with an IEEE 1900.6 CE, DA or Sensor entity, to form a SD entity:

- A spectrum database application integrates with a compliant IEEE 1900.6 entity that provides the IEEE 1900.6 logical interfaces CE/DA-S or S-S to act as a Sensor, a CE, or a DA towards any other entity that is compliant with IEEE Std 1900.6. The spectrum database application shall not modify the hosting IEEE 1900.6 entity.
- A spectrum database application modifies the hosting IEEE 1900.6 entity. The modification may include enhancements to functionality and to the particular instantiation of the IEEE 1900.6 logical interfaces CE/DA-S or S-S. Consequently, in this case, the SD entity must be denoted with a reference to its hosting IEEE 1900.6 entity, as SD(CE), SD(DA), or SD(Sensor).

NOTE 2—Instantiations of an SD logical interface might be application specific. The inclusion of a reference to the hosting entity allows distinguishing between an (assumed) SD-CE/DA or SD-S logical interface as an individual one and, for example, an SD(Sensor)-S logical interface as a reference to an SD entity assuming the role of an IEEE 1900.6 compliant Sensor entity when communicating with a Sensor that is compliant with IEEE Std 1900.6 compliant Sensor. Consequently, the SD-SD logical interface is newly introduced in the scope of scenarios that rely on the communication of spectrum sensing related information between multiple instances of an SD.

The following instances of the IEEE 1900.6 logical interfaces are involved in this scenario:

- CE/DA-S interface
- CE-CE/DA interface
- S-S interface
- SD-SD interface

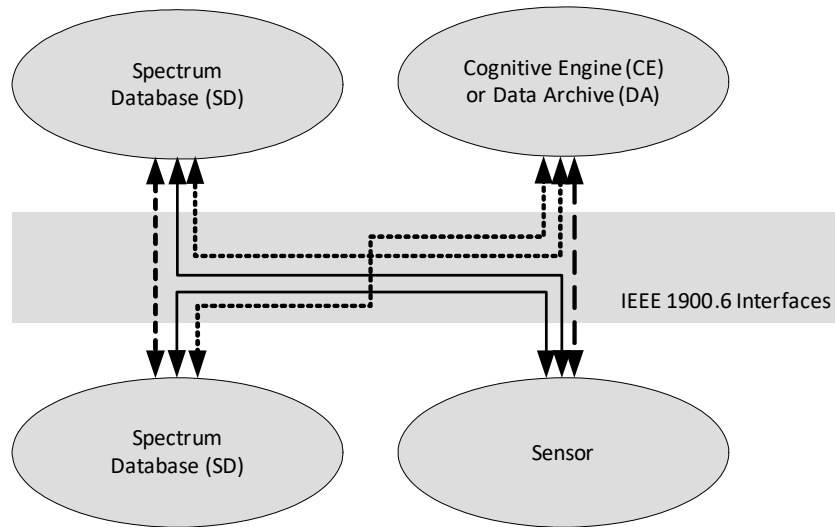
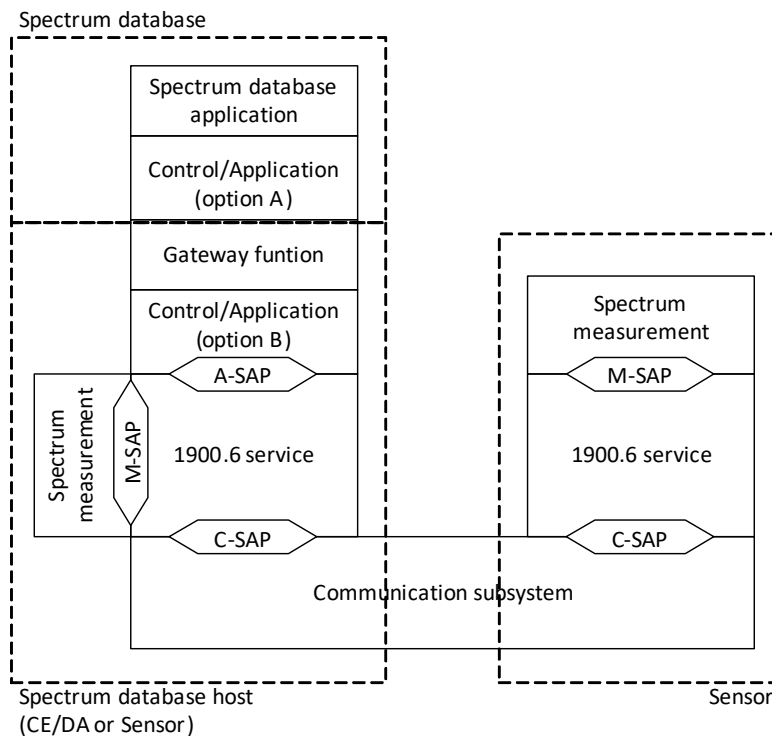


Figure 3c—Integrated Spectrum Database (SD) acting as an IEEE 1900.6 compliant entity

5. The IEEE 1900.6 reference model

5.1 General description

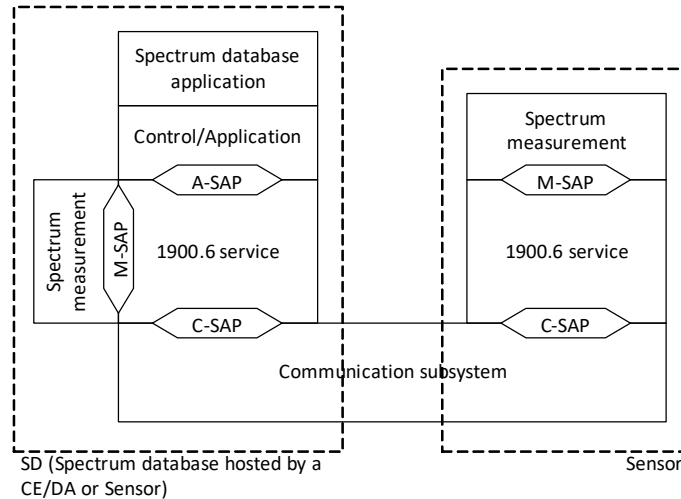
Insert the following at the end of 5.1:



NOTE 1—The IEEE 1900.6 Control/Application can be provided as part of the IEEE 1900.6 entity (option B) or as part of the spectrum database application (option A).

NOTE 2—The M-SAP is not present for CE/DA entities acting as a spectrum database host.

Figure 8b—View of the reference model for sensing related information exchange between the spectrum database and a CE/DA or Sensor through a gateway function



NOTE—This corresponds to Figure 8b option A, omitting the gateway function by integrating the spectrum database application with the hosting IEEE 1900.6 entity.

Figure 8c—View of the reference model for sensing related information exchange between the spectrum database and CE/DA or a Sensor integrating spectrum database application and IEEE 1900.6 entities

The (spectrum database) gateway function enables access of a spectrum database application to the IEEE 1900.6 service through the IEEE 1900.6 A-SAP.

The gateway function shall be implemented through one of the following options:

- As a client to the A-SAP, implementing transparent access to all A-SAP primitives. The gateway function may implement a translation of A-SAP primitives between an A-SAP and proprietary interface, but shall not apply semantic translations or functional enhancements to the A-SAP.
- As a client to A-SAP, implementing the IEEE 1900.6 Control/Application as a sub-function. The gateway function may implement additional control functions offered through the exposed interface provided that these do not interfere with the control functions required by this standard.

NOTE 1—An IEEE 1900.6 Control/Application relies on the availability of all A-SAP primitives to communicate with the IEEE 1900.6 service of an IEEE 1900.6 entity, regardless of if it accesses these through a native A-SAP (i.e., is implemented in scope of the IEEE 1900.6 instance) or if it accesses the IEEE 1900.6 service through an exposed interface (i.e., is implemented in scope of the spectrum database application). Therefore, any modifications to the A-SAP and its primitives are avoided.

The spectrum database application represents that functionality of a spectrum database that interacts with the spectrum sensing functions, or controls a distributed spectrum sensing system. It shall comply with one of the following options:

- As a client to a proprietary interface implemented by the gateway function. This interface is out of scope for this standard. This implementation instantiates option B (cf. Figure 8b) of the IEEE 1900.6 Control/Application since the gateway function is not required to provide transparent access to all A-SAP primitives.
- As a client to the IEEE 1900.6 A-SAP. This implementation instantiates option A (cf. Figure 8b) of the IEEE 1900.6 Control/Application and requires the gateway function to provide transparent access to all A-SAP primitives.

- As a client to a proprietary interface implemented by the gateway function that instantiates transparent access to all A-SAP primitives and implements proprietary services as an enhancement to the IEEE 1900.6 A-SAP. This implementation instantiates either option B (cf. Figure 8b) of the IEEE 1900.6 Control/Application to provide any functionality required by this standard, or option A for any enhanced or application-specific functionality, or might in some scenarios provide them both together.

The spectrum database application may be integrated with an IEEE 1900.6 entity as the sole client to the A-SAP according to Figure 8d. This provision does not require a hosting entity and thus does not provide any Sensor, CE, or DA functionality. In this provision, the spectrum database control and application implementation are known as the IEEE 1900.6 SD entity. As for any other IEEE 1900.6 entity, the IEEE 1900.6 service includes entity-specific functionality that is considered implementation-specific, and thus is not covered by this standard.

NOTE 2—This standard does not exclude any SD entity implementation that provides IEEE 1900.6 service functionality allowing direct C-SAP access to a spectrum database application. However, such direct C-SAP access can cause unwanted vulnerability of the distributed sensing system through the communication subsystem, leading to potential security issues.

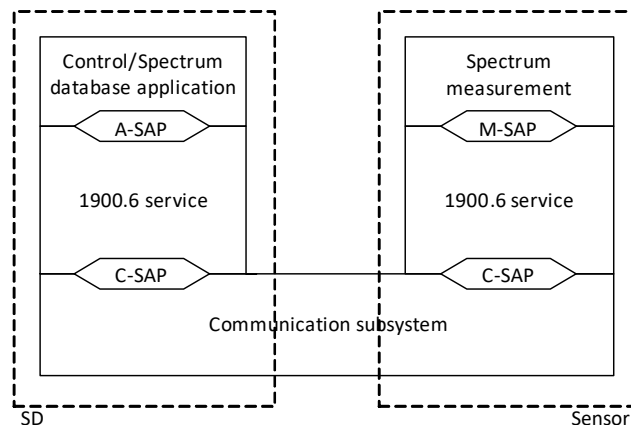


Figure 8d—View of reference model for sensing related information exchange between spectrum database (SD) and Sensor

Insert the following new subclauses after 5.1:

5.1a Security requirements

Security requirements in the reference model are provided within the A-SAP and C-SAP with the association between a certificate and single entity ID being mandatory.

NOTE 1—This is because there might be a list (entity list IDs) with changing participants represented as an entityID and the certificate is bound to the list even though the members of the list may change.

For exchanging credentials, updating, or for the creation of temporary SAPs and the generation of birth certificates, the primitives described in the following subclauses can be used to achieve these tasks securely through a secondary secure channel separating the trust checks for the information and the trust checks for the information originator.

Also, there are multiple mechanisms available in the system enabling the authorized users to change those certifications as well as the secondary channels at any point in time several times a day. For the cases of temporary IDs, these can be generated along with a certificate. If a temporary ID does not have a certificate, then it can be classified as an untrusted system.

NOTE 2—Certificates should be exchanged quite frequently. These, in conjunction with the input from the community, will form a “community of trust” model.

5.1b Entity identification method

This standard relies on identifying logical entities and instances of these by methods independent of the type of the underlying communication subsystem. SensorLogID, DataSourceLogID and ListOfDataSourceLogIDs are available, but their use is discouraged in favor of the LogicalID. Implementations of this standard shall use LogicalID, and if required should make use of the conversion methods defined in this subclause.

The LogicalID adopts the universal unique identifier (UUID) version 4 (random UUID) and version 5 (name-based SHA-1) variants according to IETF RFC 4122. The size of a LogicalID thus is 16 octets, 128 bits.

- Random UUIDs (version 4) is used for temporary LogicalIDs that have a limited lifetime or being of local relevance, such as data sources associated with a DA or CE when configuring a sensing topology. An implementation may define a particular method to generate pseudo-random values that are known in scope of a particular context (i.e., known to particular IEEE 1900.6 entities) and used within the same context such that pseudo-randomly generated LogicalIDs are deterministic in this particular context.
- Name-based UUIDs (version 5) is used in case LogicalIDs must be globally unique, such as required for sensing infrastructures shared across multiple tenants. LogicalIDs generated from the same name in the same namespace are equal with very high probability. Valid names and namespaces may be URNs, ISO LogicalIDIdentifiers, a MAC address as defined in IEEE Std 802®.

A generated LogicalID is globally unique (with high probability) and is weakly random (i.e., random but predictable, depending on the quality of the underlying random number source) and thus is not used for communication security purposes).

LogicalIDs is used if global significance and uniqueness of entity or instance identifiers are required. Shorter forms such as 8-bit, 16-bit, 32-bit unsigned integers, EUI-48 or EUI-64 identifiers, or hash values of SensorLogIDs may be used if local significance or global significance in a limited scope is sufficient.

Methods is provided to convert between LogicalIDs and shorter forms of addresses of lesser significance and vice-versa.

NOTE 1—This conversion can be reversible depending on the requirements of an implementation. Some options are discussed in Annex DA.6.

NOTE 2—An entity instance address (e.g., a device or network address) can be different from an entity identifier but may be identical if required by a particular implementation. Due to its technology-agnostic approach, this standard relies on the assumption that entity identifiers and entity addresses are separate, also assuming that entity addresses are only known to the communication subsystem under consideration.

For use as an entity address in arbitrary topologies, all instances (including permanent and temporary instances) of an IEEE 1900.6 logical entity shall receive a unique address.

- If applied to the device instantiating the IEEE 1900.6 entity, this shall be either a short form identity or a (name-based or random) LogicalID defined according to this subclause. If no LogicalID is provided, the device may generate a name-based LogicalID whenever global relevance is required.
- If assigned dynamically, the device or any coordinating device in scope (e.g., a CE instance coordinating multiple Sensor instances and communicating with those) shall generate a name-based LogicalID according to this subclause whenever global relevance is needed.

- The scope under consideration shall be defined in a way that ensures that the name used for generating a name-based LogicalID is unique within this scope. If this cannot be ensured for the given topology, a random LogicalID shall be used to avoid multiple uses of the same LogicalID in that scope.
- The namespace used for generating the LogicalID shall be explicitly or implicitly set by the instance that initiates or manages the creation of a particular sensing topology. This instance can be another IEEE 1900.6 entity or some external management function.

Each instance of a SAP has at least one unique LogicalID being unique in the scope of the SAP user and the particular context in that the SAP user is utilizing the SAP to communicate with remote entities. SAP instances may be created as a temporary instance if required and may be used in the context of entity lists (cf. 5.3), which may denote the list as the scope.

NOTE 3—For further information on referencing entity lists, refer to Annex DB.

A default SAP instance shall be provided by an instance of an IEEE 1900.6 logical entity at instantiation time and shall receive the same LogicalID as the entity, as defined in 7.5.1.

NOTE 4—In case an IEEE 1900.6 entity is shared across distinct scopes (e.g., referenced by different spectrum databases or being member of multiple entity lists), the same logical entity can have two distinct LogicalIDs, each associated with the same device and optionally with a distinct SAP instance.

5.3 Service access points

Change subclause 5.3.1 as follows:

5.3.1 Measurement service access point

The M-SAP provides a set of generic primitives or method for IEEE 1900.6 SAP users to control spectrum sensing and to obtain sensing-related information from spectrum sensing.

The IEEE 1900.6 logical entities obtain the following services from the M-SAP:

- Measurement capabilities discovery services
- Measurement configuration discovery services
- Measurement configuration services
- Management and configuration services
- Information services

Insert the following new subclause after 5.3.1.3:

5.3.1.3a Management and configuration services

5.3.1.3a.1 FeatureLockCapabilities.request

Function

This primitive requests any entity (i.e., logical entity, implementation of a service access point or subsystem) identified by a LogicalID to respond with a set of information elements that describe the entity's capabilities in realizing a lock. The LogicalID shall unambiguously identify the entity in the scope.

NOTE—The term “scope” is referring to the fact that a LogicalID can be valid in global scope (i.e., can be globally unique) or in local scope only. A local scope may be provided on a temporal or geographical basis (i.e., a LogicalID is

meaningful only during a particular period of time or at a certain geographical location), or in a certain topology (e.g., valid in the scope of another LogicalID representing the enclosing namespace).

The information returned by this function can be used by a client to lock the entity or functions provided by this entity for a given time span or for periodic occurrences thereof for exclusive use by this client.

Semantics of the service primitive

FeatureLockCapabilities.request

```
(
  TargetEntityID,
  ClientEntityID,
  ClientCERT
)
```

Parameters

Name	Type	Description
TargetEntityID	LogicalID	The identity of the entity that unambiguously identifies the target of the primitive in the scope.
ClientEntityID	LogicalID	Logical ID of the entity issuing the request.
ClientCERT (optional)	CERT	Client certificate matching the ClientEntityID given. If missing, the target entity assumes a non-authenticated requestor and may use other methods to query client authorization (e.g., requesting the local SAP instance).

When used

The primitive is used to request a remote entity to report on its capabilities for implementing a SAP feature lock. The entity issuing the request is required to invoke this primitive a prerequisite to generate subsequent lock requests. The primitive may be used by an authorized client to acquire information on the current and scheduled lock status as well as on resources already allocated to these locks.

Effect of receipt

Upon receipt, the remote entity shall report to the requestor on its lock capabilities and may report on its current and scheduled locks and affected local resources to authorized clients. The remote entity shall determine the authorization of the requesting entity by querying its local SAP instance and may utilize complementary procedures to verify this authorization.

5.3.1.3a.2 FeatureLockCapabilities.response

Function

The primitive is used in response to a FeatureLockCapabilities.request primitive received from a remote entity. The local entity replies to this request by issuing a set of mandatory and optional information elements. The response shall reflect the capabilities the entity can provide to the requester at the time of receiving the corresponding request. The entity may apply a resource reservation mechanism to reduce the probability of subsequent FeatureLock.request primitive failures.

Semantics of the service primitive

FeatureLockCapabilities.response

```
(
  ClientEntityID
  Status,
  TargetEntityID,
  TBase,
  TRes,
  MinPeriod,
  MaxWindow,
  LockSetOpt,
  LoadIndicator
)
```

Parameters

Name	Type	Description
ClientEntityID	LogicalID	The identity of the entity that issued the corresponding request primitive. This is the target to which the response is directed.
TargetEntityID	LogicalID	The identity of the entity that denotes the target of the primitive in the scope.
Status	Enumeration	Primitive return status and error code. 0: success 1: FeatureLock not supported remaining parameters is ignored 2: resource failure the entity is not able to implement a FeatureLock at the time of request. 3: authorization failure the requester has insufficient privileges to issue a FeatureLockCapabilities request—remaining parameters is ignored.
TBase	64-bit unsigned integer	System parameter—current target entity time base in ticks.
Tres	32-bit unsigned integer	System parameter—duration of a target time base tick in nanoseconds.
MinPeriod	64-bit unsigned integer	Minimum permissive lock duration in ticks (target system resource limit).
MaxWindow	64-bit unsigned integer	Maximum duration for lock windows in ticks (target system resource limit).
LockSetOpt	32-bit unsigned integer	A string of 32 bits corresponding to 32 possible LockSet parameter formats. A bit set to 1 (one, true) denotes that the corresponding LockSet parameter format is supported, a bit set to 0 (zero, false) indicates that the corresponding LockSet parameter format is not supported. The bit position (0 ... 31) refers to the LockSet format index given by the following table.

Name	Type	Description
LoadIndicator	Structured	System performance parameter indicating the current system load and resource allocations. The structure returned consists of 5 integer values of 64 bits each and a sequence (array) of LockSet parameters currently active or becoming active that are queued by the entity. 0: Earliest time a lock will apply in target time base ticks based on the LockSet parameters the entity has queued so far (64-bit unsigned integer). 1: Latest time a lock will apply in target time base ticks based on the LockSet parameters the entity has queued so far (64-bit unsigned integer). 2: Largest timespan the entity is not locked during the period defined by “latest lock time” - “earliest lock time” (64bit signed integer). Note that due to a potential wrap-around of the entity’s timer the value returned might be negative. 3: Accumulated timespan the entity is locked in target time base ticks (64-bit unsigned integer). 4: Accumulated timespan the entity is not locked in target time base ticks (64-bit unsigned integer). 5: Array of LockSet structures currently accepted and queued by the entity. The sequence does not have a specific order. Note that this parameter is present only for authorized clients (i.e., array length greater or equal to zero). For all other clients the array of LockSet parameters is empty (i.e., array length equal to zero).

LockSet parameter formats

Format Index	Scope or Feature	Description
0	Entity	Permanent Lock—the lock is active until the next LockBreak request, or until the maximum lock duration expires as given by the MaxWindow parameter. NOTE—This form is the functional equivalent of the A-SAP Lock.request primitive (see 5.3.3.3.1).
1	Entity	Timed Lock—the lock is active for a given time span and is released when the time period expires or when a LockBreak is requested.
2	Entity	Periodic Lock—the lock is active for a given time span, is released when the time period expires, and is renewed after a given idle time. The lock is released after the cycling time period expires or when a LockBreak is requested.
3 ... 31		Reserved

When used

The primitive is used in response to a FeatureLockCapabilities.request primitive to inform the requesting entity about the target entity’s capabilities to implement a FeatureLock.

Effect of receipt

Upon receipt, the requesting entity is enabled to request a FeatureLock using suitable parameter settings that reduce the probability of a request failure.

5.3.1.3a.3 FeatureLock.request

Function

This primitive requests a feature lock described by its LockSet parameter. A feature lock enables reservation of a remote entity’s resource or function for exclusive use by a requesting client. The request is directed to a

target entity (TargetEntityID), and applies to a resource or function of this entity denoted, for example, by an EntityID parameter depending on the form of the LockSet parameter used. The LockSet parameter describing the FeatureLock requested is a structured parameter having a variant layout depending on the target resource or function to which it applies. The layout format is unambiguously defined by the SAP addressed (A-SAP, C-SAP, or M-SAP) and by the format index, which is considered valid in the scope of the SAP type and its instance identified by a particular LogicalID.

If successful, the requesting client identified by the ClientEntityID parameter takes ownership of the FeatureLock.

Semantics of the service primitive

FeatureLock.request

```
(
  TargetEntityID,
  ClientEntityID,
  LockSet,
  LockPriority,
  ClientCERT
)
```

Parameters

Name	Type	Description
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive.
ClientEntityID	LogicalID	Logical ID of the entity requesting the FeatureLock. Note that this LogicalID can be different from the LogicalID of the entity issuing the request primitive (e.g., in hierarchical topologies using local controllers).
LockSet	Structured	The parameter set describing the FeatureLock requested.
LockPriority (optional)	16-bit unsigned integer	An integer parameter in the range 0 (lowest) to 65535 (highest) describing the priority of the FeatureLock requested in the scope of a given ClientEntityID.
ClientCERT (optional)	CERT	Client certificate matching the ClientEntityID given. If missing, the target entity assumes a non-authenticated requester and may use other methods to query client authorization (e.g., consulting the local SAP instance).

LockSet request parameter formats

Format Index	Scope or Feature	Format
0	Entity	LockSet (EntityID) Permanent Lock
1	Entity	LockSet (EntityID, TStart, TWindow, TolStart) Timed Lock
2	Entity	LockSet (EntityID, TStart, TWindow, Period, Cycle, TolStart, TolPeriod, TolCycle) Periodic lock
3 ... 31		Reserved (Note that format indices 28 to 31 are reserved for LockSets used in response primitives).

LockSet request parameters

Name	Type	Applies to LockSet form	Description
EntityID	Enumeration	0, 1, 2	ID of the entity within the scope of TargetEntityID to which the FeatureLock is to be applied.
TStart	64-bit unsigned integer	1, 2	Absolute time in TargetEntityID time base ticks when the FeatureLock will become active. In case of a periodic FeatureLock, the time when the FeatureLock will become active the first time.
TWindow	64-bit unsigned integer	1, 2	Duration of the FeatureLock in TargetEntityID time base ticks. In case of a periodic FeatureLock, the lifetime of the FeatureLock over all active and inactive periods. The target entity may implement the TWindow parameter to be set to multiples of a period duration (e.g., rounded up or down) ending with an active period of a FeatureLock of the duration given by the Period parameters.
Period	64-bit unsigned integer	2	The active period duration of a periodic FeatureLock in TargetEntityID time base ticks.
Cycle	64-bit unsigned integer	2	The total number of TargetEntityID time base ticks allocated to all active periods of a periodic FeatureLock. The parameter determines the FeatureLock duty cycle and shall be less or equal to the value of the TWindow parameter.
TolStart	32-bit unsigned integer	1, 2	Tolerance in TargetEntityID time base ticks for the TStart parameter as required by the FeatureLock requester. The target system may adjust the TStart parameter requested by max. $\pm$ TolStart ticks to accommodate its internal resource management requirements without informing the requester.
TolPeriod	32-bit unsigned integer	2	Tolerance in TargetEntityID time base ticks for the Period parameter as required by the FeatureLock requester. The target system may adjust the Period parameter requested by max. $\pm$ TolPeriod ticks to accommodate its internal resource management requirements without informing the requester.
TolCycle	32-bit unsigned integer	2	Tolerance in TargetEntityID time base ticks for the Cycle parameter as required by the FeatureLock requester. The target system may adjust the Cycle parameter requested by max. $\pm$ TolCycle ticks to accommodate its internal resource management requirements without informing the requester.

M-SAP FeatureLock target entities

Entity ID	Name	Applies to LockSet form	Description
0	LockRoot	0, 1, 2	The root element. The FeatureLock applies to the entity defined by TargetEntityID, usually its SAP instance. In consequence, the entity is reserved for exclusive use by the lock owner.
1	LockPwr	0, 1, 2	All entity functions that can be locked for the purpose of energy saving. Applying a lock to this entity will disable the entities sensing capabilities and, in contrast to applying locks to LockRoot, may also disable the lock owner to access the entity during the timespan a lock applies.
2	LockAcq	0, 1, 2	Acquisition of sensing parameters by the measurement module. A lock applies to the acquisition process in whole of the measurement module.
3	LockCom	0, 1, 2	Reporting of sensing parameters by the measurement module. A lock applies to the sensing data reporting eventually freeing up due communication resources or communication time slots.
4 ... 63	Reserved		
64	LockDev0	Device specific lockable entity as specified by the device's measurement module manufacturer.	
...			
64 + N	LockDevN		

When used

The primitive is used to apply a temporary lock for exclusive use of a resource or feature provided by the target. The feature the lock should be applied to can be the entity, one or more of its service access points, particular primitives of these service access points or particular parameters of such primitives.

Effect of receipt

Upon receipt, the target entity shall perform a check if the requesting entity is permitted to issue such request, shall check the validity and sanity of the primitive's parameters, and shall verify if the target system can implement the requested lock regarding requester's permissions and target entities system resources. Additionally, the target entity may evaluate the impact on the system's resource already associated with other FeatureLocks granted to other requesters and may schedule system resource reservations.

5.3.1.3a.4 FeatureLock.response**Function**

The primitive is used in response to a FeatureLock.request primitive received from a remote entity. The local entity replies by providing operation status information and a unique identifier for a FeatureLock successfully implemented to the requesting entity. The identifier of the FeatureLock can be used by the requesting entity (i.e., the FeatureLock owner) to refer to this particular FeatureLock in the course of later operations on FeatureLocks. The FeatureLock identifier is valid until the FeatureLock expires.

The local entity may or may not send an indication to the FeatureLock owner upon FeatureLock expiry using the same LockSet parameter as used in scope of this primitive after the FeatureLock expires.

Semantics of the service primitive

FeatureLock.response

```
(
  ClientEntityID,
  TargetEntityID,
  LockSet
)
```

Parameters

Name	Type	Description
ClientEntityID	LogicalID	Logical ID of the entity requesting the FeatureLock (FeatureLock owner). Note that this LogicalID can be different from the LogicalID of the entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive. Note that this LogicalID can be different from the LogicalID of the target entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
LockSet	Structured	The parameter set describing the FeatureLock status.

LockSet response parameter formats

Format Index	Scope or Feature	Format
31	All	LockSet (LockStatus, LockSetID)

LockSet response parameters

Name	Type	Format Index	Description
LockStatus	Enumeration	31	Primitive return status and error code. 0: success 1: FeatureLock not supported remaining parameters is ignored 2: resource failure the entity is not able to implement a FeatureLock at the time of request. 3: authorization failure the requester has insufficient privileges to issue a FeatureLock request. 4: FeatureLock request has been scheduled for later implementation by the local entity. 5: Invalid LockSet EntityID or FeatureLock not supported for this EntityID. 6: Resource failure when attempting to install a FeatureLock defined by the LockSet. 7: Authorization failure when attempting to install a FeatureLock defined by the LockSet.
LockSetID	LogicalID	31	A temporary LogicalID that unambiguously identifies the FeatureLock instance in the scope of the target entity. The LockSetID is created by the target entity upon successful creation or scheduling of the FeatureLock instance, and is automatically invalidated upon expiry of the FeatureLock. In all other cases, the LockSetID is invalid and is ignored by a client.

When used

The primitive is used to respond to a FeatureLock request to indicate the result of the request back to the requester. Upon successful completion it also provides a unique LockSetID that can be used as a reference identifier in subsequent operations on a FeatureLock.

Effect of receipt

Indicates a failure cause if the target entity was not able to instantiate or schedule the requested FeatureLock or provides a reference identifier to the requestor to be used in subsequent operations on the FeatureLock.

5.3.1.3a.5 FeatureUnlock.request**Function**

The primitive requests to remove a FeatureLock instance from the target entity's list of instantiated or scheduled FeatureLocks.

If scheduled but not yet active (e.g., if TStart has not been approached), the target entity shall invalidate and remove the FeatureLock immediately.

If already active, the target entity shall invalidate the reference to the FeatureLock provided by the FeatureLock.response primitive and shall terminate the FeatureLock immediately if an active lock period is not in progress, or after completion of a currently active lock period otherwise. The client may also request that an active lock period be terminated and immediate removal of the FeatureLock by setting the UnlockPriority parameter to a value greater than 0.

A requester in general can only unlock features locked if it is the authorized owner of the FeatureLock. That is, the ClientEntityID is that of the FeatureLock owner and the LockSetID is known to this client.

An authorized administrative user can unlock all FeatureLocks of an addressed target entity.

Semantics of the service primitive

FeatureUnlock.request

```
(
  TargetEntityID,
  ClientEntityID,
  LockSetID,
  UnlockPriority,
  ClientCERT
)
```

Parameters

Name	Type	Description
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive. Note that this LogicalID can be different from the LogicalID of the target entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
ClientEntityID	LogicalID	LogicalID of the entity requesting the FeatureUnlock (FeatureLock owner or authorized administrative client).
LockSetID	LogicalID	LogicalID unambiguously identifying the FeatureLock requested to unlock.
UnlockPriority	16-bit unsigned integer	An integer parameter in the range 0 (lowest) to 65535 (highest) describing the priority of the FeatureUnlock request. All values other than 0 request immediate removal of the FeatureLock (i.e., abortion).
ClientCERT (optional)	CERT	Client certificate matching the ClientEntityID given. If missing, the target entity assumes a non-authenticated requestor and may use other methods to query client authorization (e.g., consulting the local SAP instance).

When used

The primitive is issued by a client (i.e., the owner of a FeatureLock or an administrative client) unambiguously identified by its ClientEntityID to invalidate and remove a FeatureLock that has been instantiated to allocate a particular feature of the target entity for exclusive use by a client earlier.

Effect of receipt

Upon receipt, the target entity shall perform a check if the requesting entity is permitted to issue such request, shall check the validity and sanity of the primitive's parameters, and shall verify if the target system holds the implementation of the FeatureLock before attempting to cancel the FeatureLock operations and freeing up all allocated resources.

5.3.1.3a.6 FeatureUnlock.response**Function**

The primitive confirms that the FeatureLock targeted by the corresponding FeatureUnlock.request has been removed or that the request has failed. It is issued by the TargetEntityID towards the ClientEntityID.

If successful, the LockSetID is invalidated by the target entity and shall not be used any longer by the client entity, except a dedicated notification is used by the implementation to indicate deferred invalidation of FeatureLocks that have been scheduled for removal by the FeatureUnlock.request received earlier.

Semantics of the service primitive

FeatureUnlock.response

```
(
  ClientEntityID,
  TargetEntityID,
  LockSet
)
```

Parameters

Name	Type	Description
ClientEntityID	LogicalID	Logical ID of the entity requesting the FeatureUnlock (FeatureLock owner). Note that this LogicalID can be different from the LogicalID of the entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive. Note that this LogicalID can be different from the LogicalID of the target entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
LockSet	Structured	The parameter set describing the FeatureLock status.

LockSet response parameter formats

Format Index	Scope or Feature	Format
31	All	LockSet (LockStatus, LockSetID)

LockSet response parameters

Name	Type	Format Index	Description
LockStatus	Enumeration	28, 29, 30, 31	Primitive return status and error code. 0: Success, the FeatureLock has been removed 1: Success, the FeatureLock has been scheduled for removal. 2: Success, the FeatureLock has been scheduled for removal and a deferred notification will be issued upon completion of the removal process. 3: Authorization failure, the requester has insufficient privileges to issue a FeatureUnlock request. 4: Reserved. 5: Invalid LockSet EntityID or FeatureUnlock not supported for this EntityID. 6: Reserved. 7: Authorization failure when attempting to remove the FeatureLock defined by the LockSet.
LockSetID	LogicalID	28, 29, 30, 31	The LogicalID that unambiguously identifies the FeatureLock instance in the scope of the target entity. The LockSetID is created by the target entity upon successful creation or scheduling of the FeatureLock instance. Its invalidation is indicated by the reception of this primitive.

When used

The primitive is issued in response to a FeatureUnlock.request by its TargetEntityID towards the ClientEntityID (i.e., the owner of a FeatureLock or an administrative client). By this, the TargetEntityID confirms that the request has succeeded or indicates a failure.

Effect of receipt

Upon receipt, the client entity (i.e., the owner of a FeatureLock or an administrative client) shall remove any reference to the LockSet received earlier through the FeatureLock.response primitive since any further use of this reference will cause a failure response by the target entity.

5.3.1.3a.7 FeatureBreaklock.request**Function**

The primitive requests to invalidate and remove one or more FeatureLocks instantiated by the particular target client regardless of their actual ownership.

This primitive is available to authorized clients only. Implementations shall ensure that only FeatureBreaklock.requests issued by clients of higher authorization level than that of the FeatureLock owner are allowed to proceed. Note that FeatureBreaklock primitives shall not be lockable themselves.

Semantics of the service primitive

FeatureBreaklock.request

```
(
  TargetEntityID,
  ClientEntityID,
  LockSet,
  UnlockPriority,
  ClientCERT
)
```

Parameters

Name	Type	Description
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive. Note that this LogicalID can be different from the LogicalID of the target entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers or when the administrative client is not the FeatureLock owner).
ClientEntityID	LogicalID	Logical ID of the entity requesting the FeatureBreaklock (authorized administrative client).
LockSet	Structured	The parameter set describing the FeatureLock requested.
UnlockPriority	16-bit unsigned integer	An integer parameter in the range 0 (lowest) to 65535 (highest) describing the priority of the FeatureBreaklock request. All values other than 0 request immediate removal of the FeatureLock (i.e., abortion).
ClientCERT (optional)	CERT	Client certificate matching the ClientEntityID given. If missing, the target entity shall use other methods to query client authorization (e.g., consulting the local SAP instance).

LockSet request parameter formats

Format Index	Scope or Feature	Format
0	Entity	LockSet (EntityID) Remove all FeatureLocks implemented by the TargetEntityID that matches EntityID.
1 ... 25		Reserved
26	Entity	LockSet (LogicalID) Remove the FeatureLock specified by the LogicalID.
27	Entity	LockSet () Remove all FeatureLocks implemented by the TargetEntityID
28 ... 31		Reserved

LockSet request parameters

Name	Type	Applies to LockSet form	Description
EntityID	Enumeration	0	ID of the entity within the scope of TargetEntityID to which the FeatureLock is to be applied.
LockSetID	LogicalID	26	The LogicalID that unambiguously identifies the FeatureLock instance in the scope of the target entity.

M-SAP FeatureBreaklock target entities (cf. 5.3.1.3a.3)

EntityID	Name	Applies to LockSet form	Description
0	LockRoot	0, 26	The root element. The FeatureLock applies to the entity defined by TargetEntityID, usually its SAP instance. If successful, this request removes all locks currently active for TargetEntityID and is therefore functionally equivalent to LockSet form 27 omitting the TargetEntityID parameter.
1	LockPwr	0	Removes all locks of type LockPwr for this TargetEntityID.
2	LockAcq	0	Removes all locks of type LockAcq for this TargetEntityID.
3	LockCom	0	Removes all locks of type LockCom for this TargetEntityID.
4 ... 63	Reserved		
64	LockDev0	Removes all locks of type LockDev<Id> for this TargetEntityID. The effect of removing this lock is as specified by the device's measurement module manufacturer.	
...			
64 + N	LockDevN		

When used

The primitive is used primarily by administrative clients that need to free up a particular resource locked by a particular client, or to lock out a particular client's ability to keep FeatureLocks.

Effect of receipt

Upon receipt, the target entity shall perform a check if the requesting entity is permitted to issue such request, shall check the validity and sanity of the primitive's parameters, and shall verify if the target system holds the implementation of the FeatureLock before attempting to cancel the FeatureLock operations and freeing up all allocated resources.

5.3.1.3a.8 FeatureBreaklock.response**Function**

The primitive confirms that the FeatureLock targeted by the corresponding FeatureBreaklock.request has been removed or that the request has failed. It is issued by the TargetEntityID towards the ClientEntityID.

If successful, the LockSetID is invalidated by the target entity and shall not be used any longer by the client entity, except a dedicated notification is used by the implementation to indicate deferred invalidation of FeatureLocks that have been scheduled for removal by the FeatureBreaklock.request received earlier.

Semantics of the service primitive

FeatureBreaklock.response

```
(
  ClientEntityID,
  TargetEntityID,
  LockSet
)
```

Parameters

Name	Type	Description
ClientEntityID	LogicalID	Logical ID of the entity requesting the FeatureBreaklock (administrative client). Note that this LogicalID can be different from the LogicalID of the entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers or when the administrative client is not the FeatureLock owner).
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive. Note that this LogicalID can be different from the LogicalID of the target entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers or when the administrative client is not the FeatureLock owner).
LockSet	Structured	The parameter set describing the FeatureLock status.

LockSet response parameter formats

Format Index	Scope or Feature	Format
31	all	LockSet (LockStatus, LockSetID)

LockSet response parameters

Name	Type	Format Index	Description
LockStatus	Enumeration	28, 29, 30, 31	Primitive return status and error code. 0: Success, the FeatureLock has been removed 1: Success, the FeatureLock has been scheduled for removal. 2: Success, the FeatureLock has been scheduled for removal and a deferred notification will be issued upon completion of the removal process. 3: Authorization failure, the requester has insufficient privileges to issue a FeatureBreaklock request. 4: Reserved. 5: Invalid LockSet EntityID or FeatureBreaklock not supported for this EntityID. 6: Reserved. 7: Authorization failure when attempting to remove the FeatureLock defined by the LockSet.
LockSetID	LogicalID	NA	The LogicalID that unambiguously identifies the FeatureLock instance in the scope of the target entity. The LockSetID is created by the target entity upon successful creation or scheduling of the FeatureLock instance. Its invalidation is indicated by the reception of this primitive.

When used

The primitive is issued in response to a FeatureBreaklock.request by its TargetEntityID towards the ClientEntityID (i.e., the administrative client). By this, the TargetEntityID confirms that the request has been accepted or rejected.

Effect of receipt

Upon receipt, the client entity (i.e., the administrative client) shall remove any reference to the LockSetIDs under consideration since any further use of this reference will cause a failure response by the target entity.

The local entity may or may not send an indication to the FeatureLock owner upon successful removal of the FeatureLock implementation under consideration.

Insert the following after 5.3.1.4.15:

5.3.1.4.16 Notification.request

Function

This primitive is used by the M-SAP user (e.g., the IEEE 1900.6 service) to send an asynchronous status change information, exception condition information or error notification to the M-SAP instance, to the measurement module, or to any entity accessible through this M-SAP instance (e.g., if being the M-SAP of a proxy Sensor, cf. 4.4).

Semantics of the service primitive

Notification.request
(
 DestEntityID,
 SrcEntityID,

Category,
Code,
Reason,
SrcCERT
)

Parameters

Name	Type	Description
DestEntityID (optional)	LogicalID	The identity of the entity that unambiguously identifies the destination of the notification in the scope. If missing, the scope is set to the M-SAP instance that is used to issue the primitive. The SAP instance is responsible to resolve a suitable destination matching Category and Code parameters when needed.
SrcEntityID (optional)	LogicalID	Logical ID of the entity issuing the request. Note that the SAP user may have forwarded the request from another source. If missing, the SrcCERT parameter is ignored, and the request is handled on a least significant authorization level.
Category	Enumeration	Category of the request (cf. 6.3a Table 1a).
Code	Enumeration	Error or result code (cf. 6.3a Table 1a). Note that there is no constraint regarding any combination of Category and Code. All suitable combinations are valid and is documented accordingly when describing implementation dependent interface procedures.
SrcCERT (optional)	CERT	Client certificate matching the SrcEntityID given. If missing, the destination or forwarding entity assumes a non-authenticated requestor and may use other methods to query client authorization (e.g., requesting the local SAP instance). NOTE—This parameter is for further study.
Reason (optional)	String	An optional short human readable description of the cause of the notification.

When used

This primitive is used by the M-SAP user (e.g., the IEEE 1900.6 service) to inform the M-SAP instance or any particular entity accessible through this M-SAP instance about an asynchronous event, error condition, exception or state change that is not related to a particular request primitive and therefore cannot be reported through a corresponding response or is due to a deferred action associated with a particular request primitive.

Effect of receipt

Upon reception of the request and after its formal validation, the service providing the M-SAP instance (e.g., the measurement module) shall take appropriate actions to process the request, or to delegate or forward the request to a functional entity that is able to process the request in the scope.

5.3.1.4.17 Notification.indication

Function

This primitive is used by the service providing an M-SAP instance (e.g., the measurement module) to send an asynchronous status change information, exception condition information or error notification to the M-SAP user (e.g., the IEEE 1900.6 service), or to another entity accessible through this M-SAP instance (e.g., a C-SAP or A-SAP instance).

Note that, when omitting DestEntityID, SrcEntityID and SrcCERT parameters, this is the functional equivalent of the M-SAP Notify primitive (cf. 5.3.1.4.15).

Semantics of the service primitive

Notification.indication

```
(
    DestEntityID,
    SrcEntityID,
    Category,
    Code,
    Reason,
    SrcCERT
)
```

Parameters

Name	Type	Description
DestEntityID (optional)	LogicalID	The identity of the entity that unambiguously identifies the target of the primitive in the scope. If missing, the scope is set to the user of this M-SAP instance. The M-SAP user may decide to resolve a suitable destination matching Category and Code parameters when needed.
SrcEntityID (optional)	LogicalID	Logical ID of the entity issuing the request. Note that the SAP provider can forward request from other source than the M-SAP provider (e.g., the measurement module) itself. If missing, the SrcCERT parameter is ignored, and the request is handled on a least significant authorization level.
Category	Enumeration	Category of the request (cf. 6.3a Table 1a).
Code	Enumeration	Error or result code (cf. 6.3a Table 1a).
SrcCERT (optional)	CERT	Client certificate matching the SrcEntityID given. If missing, the destination or forwarding entity assumes a non-authenticated requestor and may use other methods to query client authorization (e.g., requesting the local SAP instance). NOTE—This parameter is for further study.
Reason (optional)	String	An optional short human readable description of the cause of the notification.

When used

This primitive is used by the M-SAP provider (e.g., the measurement module) to inform the M-SAP user or any particular entity accessible through this M-SAP instance about an asynchronous event, error condition, exception, or state change that is not related to a particular request primitive and therefore cannot be reported through a corresponding response or is due to a deferred action associated with a particular request primitive.

Effect of receipt

Upon reception of the indication and after its formal validation, the M-SAP user shall take appropriate actions to process the request, or to delegate or forward the request to a functional entity that is able to process the request in the scope.

5.3.2 Communication service access point

Change the text in 5.3.2 as follows:

The C-SAP is used for the exchange of sensing-related information (cf. 6.1) between Sensors and their clients. The client role can be taken by a Sensor, CE, or DA. It abstracts services of the communication by providing a set of generic primitives or method and mapping these primitives to transport protocols.

The IEEE 1900.6 SAP users obtain the following services from the C-SAP:

- Sensing-related information send service
- Sensing-related information receive service

- Management and configuration services
- Information services
- Security services

Insert the following after 5.3.2.2:

5.3.2.2a Management and configuration services

5.3.2.2a.1 FeatureLockCapabilities.request

Function

This primitive requests any entity (i.e., logical entity, implementation of a service access point or subsystem) identified by a LogicalID to respond with a set of information elements that describe the entity's capabilities in realizing a lock. The LogicalID shall unambiguously identify the entity in the scope.

NOTE—The term *scope* is referring to the fact that a LogicalID can be valid in global scope (i.e., can be globally unique) or in local scope only. A local scope may be provided on a temporal or geographical basis (i.e., a LogicalID is meaningful only during a particular period of time or at a certain geographical location), or in a certain topology (e.g., valid in the scope of another LogicalID representing the enclosing namespace).

The information returned by this function can be used by a client to lock the entity or functions provided by this entity for a given time span or for periodic occurrences thereof for exclusive use by this client.

Semantics of the service primitive

FeatureLockCapabilities.request

```
(
  TargetEntityID,
  ClientEntityID,
  ClientCERT
)
```

Parameters

Name	Type	Description
TargetEntityID	LogicalID	The identity of the entity that unambiguously identifies the target of the primitive in the scope.
ClientEntityID	LogicalID	Logical ID of the entity issuing the request.
ClientCERT (optional)	CERT	Client certificate matching the ClientEntityID given. If missing, the target entity assumes a non-authenticated requestor and may use other methods to query client authorization (e.g., requesting the local SAP instance).

When used

The primitive is used to request a remote entity to report on its capabilities for implementing a SAP feature lock. The entity issuing the request is required to invoke this primitive a prerequisite to generate subsequent lock requests. The primitive may be used by an authorized client to acquire information on the current and scheduled lock status as well as on resources already allocated to these locks.

Effect of receipt

Upon receipt, the remote entity shall report to the requestor on its lock capabilities, and may report on its current and scheduled locks and affected local resources to authorized clients. The remote entity shall determine the authorization of the requesting entity by querying its local SAP instance and may utilize complementary procedures to verify this authorization.

5.3.2.2a.2 FeatureLockCapabilities.response**Function**

The primitive is used in response to a FeatureLockCapabilities.request primitive received from a remote entity. The local entity replies to this request by issuing a set of mandatory and optional information elements. The response shall reflect the capabilities the entity can provide to the requester at the time of receiving the corresponding request. The entity may apply a resource reservation mechanism to reduce the probability of subsequent FeatureLock.request primitive failures.

Semantics of the service primitive

```
FeatureLockCapabilities.response
(
    ClientEntityID
    Status,
    TargetEntityID,
    TBase,
    TRes,
    MinPeriod,
    MaxWindow,
    LockSetOpt,
    LoadIndicator
)
```

Parameters

Name	Type	Description
ClientEntityID	LogicalID	The identity of the entity that issued the corresponding request primitive. This is the target to which the response is directed.
TargetEntityID	LogicalID	The identity of the entity that denotes the target of the primitive in the scope.
Status	Enumeration	Primitive return status and error code. 0: success 1: FeatureLock not supported remaining parameters is ignored 2: resource failure the entity is not able to implement a FeatureLock at the time of request. 3: authorization failure the requester has insufficient privileges to issue a FeatureLockCapabilities request – remaining parameters is ignored
TBase	64-bit unsigned integer	System parameter – current target entity time base in ticks.
Tres	32-bit unsigned integer	System parameter – duration of a target time base tick in nanoseconds.
MinPeriod	64-bit unsigned integer	Minimum permissive lock duration in ticks (target system resource limit).
MaxWindow	64-bit unsigned integer	Maximum duration for lock windows in ticks (target system resource limit).
LockSetOpt	32-bit unsigned integer	A string of 32 bits corresponding to 32 possible LockSet parameter formats. A bit set to 1 (one, true) denotes that the corresponding LockSet parameter format is supported, a bit set to 0 (zero, false) indicates that the corresponding LockSet parameter format is not supported. The bit position (0 ... 31) refers to the LockSet format index given by the following table.

Name	Type	Description
LoadIndicator	Structured	System performance parameter indicating the current system load and resource allocations. The structure returned consists of 5 integer values of 64 bits each and a sequence (array) of LockSet parameters currently active or becoming active that are queued by the entity. 0: Earliest time a lock will apply in target time base ticks based on the LockSet parameters the entity has queued so far (64-bit unsigned integer). 1: Latest time a lock will apply in target time base ticks based on the LockSet parameters the entity has queued so far (64-bit unsigned integer). 2: Largest timespan the entity is not locked during the period defined by “latest lock time” - “earliest lock time” (64-bit signed integer). Note that due to a potential wrap-around of the entity’s timer the value returned might be negative. 3: Accumulated timespan the entity is locked in target time base ticks (64-bit unsigned integer). 4: Accumulated timespan the entity is not locked in target time base ticks (64-bit unsigned integer). 5: Array of LockSet structures currently accepted and queued by the entity. The sequence does not have a specific order. Note that this parameter is present only for authorized clients (i.e., array length greater or equal to zero). For all other clients the array of LockSet parameters is empty (i.e., array length equal to zero).

LockSet parameter formats

Format Index	Scope or Feature	Description
0	Entity	Permanent Lock – The lock is active until the next LockBreak request, or until the maximum lock duration expires as given by the MaxWindow parameter. NOTE—This form is the functional equivalent of the A-SAP Lock.request primitive (see 5.3.3.3.1).
1	Entity	Timed Lock – The lock is active for a given time span and is released when the time period expires or when a LockBreak is requested.
2	Entity	Periodic Lock – The lock is active for a given time span, is released when the time period expires, and is renewed after a given idle time. The lock is released after the cycling time period expires or when a LockBreak is requested.
3 ... 31		Reserved

When used

The primitive is used in response to a FeatureLockCapabilities.request primitive to inform the requesting entity about the target entity’s capabilities to implement a FeatureLock.

Effect of receipt

Upon receipt, the requesting entity is enabled to request a FeatureLock using suitable parameter settings that reduce the probability of a request failure.

5.3.2.2a.3 FeatureLock.request

Function

This primitive requests a feature lock described by its LockSet parameter. A feature lock enables reservation of a remote entity’s resource or function for exclusive use by a requesting client. The request is directed to a target entity (TargetEntityID), and applies to a resource or function of this entity denoted, for example, by an EntityID parameter depending on the form of the LockSet parameter used. The LockSet parameter describing

the FeatureLock requested is a structured parameter that has a variant layout dependent on the target resource or function to which it applies. The layout format is unambiguously defined by the SAP addressed (A-SAP, C-SAP, or M-SAP) and by the format index, which is considered valid in the scope of the SAP type and its instance identified by a particular LogicalID.

If successful, the requesting client identified by the ClientEntityID parameter takes ownership of the FeatureLock.

Semantics of the service primitive

FeatureLock.request

```
(
  TargetEntityID,
  ClientEntityID,
  LockSet,
  LockPriority,
  ClientCERT
)
```

Parameters

Name	Type	Description
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive.
ClientEntityID	LogicalID	Logical ID of the entity requesting the FeatureLock. Note that this LogicalID can be different from the LogicalID of the entity issuing the request primitive (e.g., in hierarchical topologies using local controllers).
LockSet	Structured	The parameter set describing the FeatureLock requested.
LockPriority (optional)	16-bit unsigned integer	An integer parameter in the range 0 (lowest) to 65535 (highest) describing the priority of the FeatureLock requested in the scope of a given ClientEntityID.
ClientCERT (optional)	CERT	Client certificate matching the ClientEntityID given. If missing, the target entity assumes a non-authenticated requester and may use other methods to query client authorization (e.g., consulting the local SAP instance).

LockSet request parameter formats

Format Index	Scope or Feature	Format
0	Entity	LockSet (EntityID) Permanent Lock
1	Entity	LockSet (EntityID, TStart, TWindow, TolStart) Timed Lock
2	Entity	LockSet (EntityID, TStart, TWindow, Period, Cycle, TolStart, TolPeriod, TolCycle) Periodic lock
3 ... 31		Reserved (Note that format indices 28 to 31 are reserved for LockSets used in response primitives).

LockSet request parameters

Name	Type	Applies to LockSet form	Description
EntityID	Enumeration	0, 1, 2	ID of the entity within the scope of TargetEntityID to which the FeatureLock is to be applied.
TStart	64-bit unsigned integer	1, 2	Absolute time in TargetEntityID time base ticks when the FeatureLock will become active. In case of a periodic FeatureLock, the time when the FeatureLock will become active the first time.
TWindow	64-bit unsigned integer	1, 2	Duration of the FeatureLock in TargetEntityID time base ticks. In case of a periodic FeatureLock, the lifetime of the FeatureLock over all active and inactive periods. The target entity may implement the TWindow parameter to be set to multiples of a period duration (e.g., rounded up or down) ending with an active period of a FeatureLock of the duration given by the Period parameters.
Period	64-bit unsigned integer	2	The active period duration of a periodic FeatureLock in TargetEntityID time base ticks.
Cycle	64-bit unsigned integer	2	The total number of TargetEntityID time base ticks allocated to all active periods of a periodic FeatureLock. The parameter determines the FeatureLock duty cycle and shall be less or equal to the value of the TWindow parameter.
TolStart	32-bit unsigned integer	1, 2	Tolerance in TargetEntityID time base ticks for the TStart parameter as required by the FeatureLock requester. The target system may adjust the TStart parameter requested by max. $\pm$ TolStart ticks to accommodate its internal resource management requirements without informing the requester.
TolPeriod	32-bit unsigned integer	2	Tolerance in TargetEntityID time base ticks for the Period parameter as required by the FeatureLock requester. The target system may adjust the Period parameter requested by max. $\pm$ TolPeriod ticks to accommodate its internal resource management requirements without informing the requester.
TolCycle	32-bit unsigned integer	2	Tolerance in TargetEntityID time base ticks for the Cycle parameter as required by the FeatureLock requester. The target system may adjust the Cycle parameter requested by max. $\pm$ TolCycle ticks to accommodate its internal resource management requirements without informing the requester.

C-SAP FeatureLock target entities

EntityID	Name	Applies to LockSet form	Description
0	LockRoot	0, 1, 2	The root element. The FeatureLock applies to the entity defined by TargetEntityID, usually its SAP instance. In consequence, the entity is reserved for exclusive use by the lock owner. Note that this lock also applies to the corresponding remote instance of the C-SAP addressed if any.
1	LockPwr	0, 1, 2	All entity functions that can be locked for the purpose of energy saving. Applying a lock to this entity will restrict or disable the entities communication capabilities and, in contrast to applying locks to LockRoot, may also disable the lock owner to access the entity during the timespan a lock applies.
2	LockLoc	0, 1, 2	Local access to the C-SAP under consideration. A lock applies to the local instance of the C-SAP only. TargetEntityID here denotes the identity of a local lock target.
3	LockE2E	0, 1, 2	Communication path between a local C-SAP instance and the corresponding remote instance if any, including local C-SAP instance, remote instance and all intermediaries if any. TargetEntityID here denotes the remote endpoint of the communication path under consideration, which might be a SAP instance of a remote logical entity. Note that in case more than one communication path exists between local and remote endpoint, the lock shall affect access to all of them (e.g., all legs in a multi-hop architecture).
4	LockTx	0, 1, 2	Capability to transmit information through this entity. This lock applies to all send information or notification primitives provided by the C-SAP, for example.
5	LockRx	0, 1, 2	Capability to receive information through this entity. This lock applies to all receive or get information as well as notification primitives implemented by the C-SAP, for example.
6 ... 63	Reserved		
64	LockDev0	Implementation specific lockable entity as specified by the Communication subsystem's manufacturer or manager (e.g., in case of a communication network-based implementation).	
...			
64 + N	LockDevN		

When used

The primitive is used to apply a temporary lock for exclusive use of a resource or feature provided by the target. The feature the lock should be applied to can be the entity, one or more of its service access points, particular primitives of these service access points or particular parameters of such primitives.

Effect of receipt

Upon receipt, the target entity shall perform a check if the requesting entity is permitted to issue such request, shall check the validity and sanity of the primitive's parameters, and shall verify if the target system can implement the requested lock regarding requester's permissions and target entities system resources. Additionally, the target entity may evaluate the impact on the system's resource already associated with other FeatureLocks granted to other requesters and may schedule system resource reservations.

5.3.2.2a.4 FeatureLock.response**Function**

The primitive is used in response to a FeatureLock.request primitive received from a remote entity. The local entity replies by providing operation status information and a unique identifier for a FeatureLock successfully

implemented to the requesting entity. The identifier of the FeatureLock can be used by the requesting entity (i.e., the FeatureLock owner) to refer to this particular FeatureLock in the course of later operations on FeatureLocks. The FeatureLock identifier is valid until the FeatureLock expires.

The local entity may or may not send an indication to the FeatureLock owner upon FeatureLock expiry using the same LockSet parameter as used in scope of this primitive after the FeatureLock expires.

Semantics of the service primitive

FeatureLock.response

```
(
  ClientEntityID,
  TargetEntityID,
  LockSet
)
```

Parameters

Name	Type	Description
ClientEntityID	LogicalID	Logical ID of the entity requesting the FeatureLock (FeatureLock owner). Note that this LogicalID can be different from the LogicalID of the entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive. Note that this LogicalID can be different from the LogicalID of the target entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
LockSet	Structured	The parameter set describing the FeatureLock status.

LockSet response parameter formats

Format Index	Scope or Feature	Format
31	All	LockSet (LockStatus, LockSetID)

LockSet response parameters

Name	Type	Format Index	Description
LockStatus	Enumeration	28, 29, 30, 31	Primitive return status and error code. 0: success 1: FeatureLock not supported remaining parameters is ignored 2: resource failure the entity is not able to implement a FeatureLock at the time of request. 3: authorization failure the requester has insufficient privileges to issue a FeatureLock request. 4: FeatureLock request has been scheduled for later implementation by the local entity. 5: Invalid LockSet EntityID or FeatureLock not supported for this EntityID. 6: Resource failure when attempting to install a FeatureLock defined by the LockSet. 7: Authorization failure when attempting to install a FeatureLock defined by the LockSet.
LockSetID	LogicalID	28, 29, 30, 31	A temporary LogicalID that unambiguously identifies the FeatureLock instance in the scope of the target entity. The LockSetID is created by the target entity upon successful creation or scheduling of the FeatureLock instance, and is automatically invalidated upon expiry of the FeatureLock. In all other cases, the LockSetID is invalid and is ignored by a client.

When used

The primitive is used to respond to a FeatureLock request to indicate the result of the request back to the requester. Upon successful completion, it also provides a unique LockSetID that can be used as a reference identifier in subsequent operations on a FeatureLock.

Effect of receipt

Indicates a failure cause if the target entity was not able to instantiate or schedule the requested FeatureLock, or provides a reference identifier to the requestor to be used in subsequent operations on the FeatureLock.

5.3.2.2a.5 FeatureUnlock.request**Function**

The primitive requests to remove a FeatureLock instance from the target entity's list of instantiated or scheduled FeatureLocks.

If scheduled but not yet active (e.g., if TStart has not been approached), the target entity shall invalidate and remove the FeatureLock immediately.

If already active, the target entity shall invalidate the reference to the FeatureLock provided by the FeatureLock.response primitive and shall aim to terminate the FeatureLock immediately but after completion of a currently active lock period, if applicable. The client may also request an immediate removal (i.e., abortion) of the FeatureLock by setting the UnlockPriority parameter to a suitable value.

A requester in general can only unlock features locked if it is the authorized owner of the FeatureLock. That is, the ClientEntityID is that of the FeatureLock owner and the LockSetID is known to this client.

An authorized administrative user can unlock all FeatureLocks of an addressed target entity.

Semantics of the service primitive

FeatureUnlock.request

```
(
  TargetEntityID,
  ClientEntityID,
  LockSetID,
  UnlockPriority,
  ClientCERT
)
```

Parameters

Name	Type	Description
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive. Note that this LogicalID can be different from the LogicalID of the target entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
ClientEntityID	LogicalID	LogicalID of the entity requesting the FeatureUnlock (FeatureLock owner or authorized administrative client).
LockSetID	LogicalID	LogicalID unambiguously identifying the FeatureLock requested to unlock.
UnlockPriority	16-bit unsigned integer	An integer parameter in the range 0 (lowest) to 65535 (highest) describing the priority of the FeatureUnlock request. All values other than 0 request immediate removal of the FeatureLock (i.e., abortion).
ClientCERT (optional)	CERT	Client certificate matching the ClientEntityID given. If missing, the target entity assumes a non-authenticated requestor and may use other methods to query client authorization (e.g., consulting the local SAP instance).

When used

The primitive is issued by a client (i.e., the owner of a FeatureLock or an administrative client) unambiguously identified by its ClientEntityID to invalidate and remove a FeatureLock that has been instantiated to allocate a particular feature of the target entity for exclusive use by a client earlier.

Effect of receipt

Upon receipt, the target entity shall perform a check if the requesting entity is permitted to issue such request, shall check the validity and sanity of the primitive's parameters, and shall verify if the target system holds the implementation of the FeatureLock before attempting to cancel the FeatureLock operations and freeing up all allocated resources.

5.3.2.2a.6 FeatureUnlock.response**Function**

The primitive confirms that the FeatureLock targeted by the corresponding FeatureUnlock.request has been removed or that the request has failed. It is issued by the TargetEntityID towards the ClientEntityID.

If successful, the LockSetID is invalidated by the target entity and shall not be used any longer by the client entity, except a dedicated notification is used by the implementation to indicate deferred invalidation of FeatureLocks that have been scheduled for removal by the FeatureUnlock.request received earlier.

Semantics of the service primitive

FeatureUnlock.response

```
(
  ClientEntityID,
  TargetEntityID,
  LockSet
)
```

Parameters

Name	Type	Description
ClientEntityID	LogicalID	Logical ID of the entity requesting the FeatureUnlock (FeatureLock owner). Note that this LogicalID can be different from the LocalID of the entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
TargetEntityID	LogicalID	The identity of the entity that is the target in the scope of the primitive. Note that this LogicalID can be different from the LocalID of the target entity when issuing the request primitive (e.g., in hierarchical topologies using local controllers).
LockSet	Structured	The parameter set describing the FeatureLock status.

LockSet response parameter formats

Format Index	Scope or Feature	Format
31	All	LockSet (LockStatus, LockSetID)

LockSet response parameters

Name	Type	Format Index	Description
LockStatus	Enumeration	28, 29, 30, 31	Primitive return status and error code. 0: Success, the FeatureLock has been removed 1: Success, the FeatureLock has been scheduled for removal. 2: Success, the FeatureLock has been scheduled for removal and a deferred notification will be issued upon completion of the removal process. 3: Authorization failure, the requester has insufficient privileges to issue a FeatureUnlock request. 4: Reserved. 5: Invalid LockSet EntityID or FeatureUnlock not supported for this EntityID. 6: Reserved. 7: Authorization failure when attempting to remove the FeatureLock defined by the LockSet.
LockSetID	LogicalID	28, 29, 30, 31	The LogicalID that unambiguously identifies the FeatureLock instance in the scope of the target entity. The LockSetID is created by the target entity upon successful creation or scheduling of the FeatureLock instance. Its invalidation is indicated by the reception of this primitive.

When used

The primitive is issued in response to a FeatureUnlock.request by its TargetEntityID towards the ClientEntityID (i.e., the owner of a FeatureLock or an administrative client). By this, the TargetEntityID confirms that the request has succeeded or indicates a failure.

Effect of receipt

Upon receipt, the client entity (i.e., the owner of a FeatureLock or an administrative client) shall remove any reference to the LockSet received earlier through the FeatureLock.response primitive since any further use of this reference will cause a failure response by the target entity.