P802.22™/D1  
Draft Standard for Information Technology – Telecommunications and information exchange between systems  
Wireless Regional Area Networks (WRAN)—Specific requirements  
  
Part 22.3: Spectrum Characterization and Occupancy Sensing

Sponsor

LAN/MAN Standards Committee

of the

IEEE Computer Society

Approved <Date Approved>

IEEE-SA Standards Board

Copyright © 2014 by The Institute of Electrical and Electronics Engineers, Inc.

Three Park Avenue

New York, New York 10016-5997, USA

All rights reserved.

This document is an unapproved draft of a proposed IEEE Standard. As such, this document is subject to change. USE AT YOUR OWN RISK! IEEE copyright statements SHALL NOT BE REMOVED from draft or approved IEEE standards, or modified in any way. Because this is an unapproved draft, this document must not be utilized for any conformance/compliance purposes. Permission is hereby granted for officers from each IEEE Standards Working Group or Committee to reproduce the draft document developed by that Working Group for purposes of international standardization consideration. IEEE Standards Department must be informed of the submission for consideration prior to any reproduction for international standardization consideration ([stds.ipr@ieee.org](mailto:stds.ipr@ieee.org)). Prior to adoption of this document, in whole or in part, by another standards development organization, permission must first be obtained from the IEEE Standards Department ([stds.ipr@ieee.org](mailto:stds.ipr@ieee.org)). When requesting permission, IEEE Standards Department will require a copy of the standard development organization's document highlighting the use of IEEE content. Other entities seeking permission to reproduce this document, in whole or in part, must also obtain permission from the IEEE Standards Department.

IEEE Standards Department

445 Hoes Lane

Piscataway, NJ 08854, USA

Abstract: This standard specifies the architecture, abstraction layers, interfaces and metadata requirements for Spectrum Characterization and Occupancy Sensing (SCOS) system, and defines performance parameters, units and measures. This SCOS system comprises one or more semi-autonomous Spectrum Sensing Devices which scan electromagnetic spectrum, digitize it and perform processing, transmitting the resultant data with appropriate metadata to a central storage and processing system, according to rules, policies or instructions imposed on the Spectrum Sensing Devices by a management system.

Keywords: radio spectrum sensing, spectrum monitoring, signal characterization, cognitive radio, IEEE 802.22.3, WRAN standards

[[1]](#footnote-1)•

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading “Important Notices and Disclaimers Concerning IEEE Standards Documents.” They can also be obtained on request from IEEE or viewed at <http://standards.ieee.org/IPR/disclaimers.html>.

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

IEEE Standards documents (standards, recommended practices, and guides), both full-use and trial-use, are developed within IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (“IEEE-SA”) Standards Board. IEEE (“the Institute”) develops its standards through a consensus development process, approved by the American National Standards Institute (“ANSI”), which brings together volunteers representing varied viewpoints and interests to achieve the final product. IEEEStandards are documents developed through scientific, academic, and industry-based technical working groups. Volunteers in IEEE working groups are not necessarily members of the Institute 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 Standards do not guarantee or ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. 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.

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. 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: PROCUREMENT OF 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 should be considered 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, or 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 his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

**Comments on standards**

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide 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 Standards Coordinating Committees 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 revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE**-**SA Standards Board

445 Hoes Lane

Piscataway, NJ 08854 USA

**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 imply 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.

**Copyrights**

IEEE draft and approved standards are copyrighted by IEEE under U.S. 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 by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

**Photocopies**

Subject to payment of the appropriate fee, 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. 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 ten years. When a document is more than ten 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 the IEEE Xplore at <http://ieeexplore.ieee.org/> or contact IEEE at the address listed previously. For more information about the IEEE-SA or IEEE’s standards development process, visit the IEEE-SA Website at <http://standards.ieee.org>.

**Errata**

Errata, if any, for all IEEE standards can be accessed on the IEEE-SA Website at the following URL: <http://standards.ieee.org/findstds/errata/index.html>. Users are encouraged to check this URL for errata periodically.

**Patents**

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 <http://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.

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.

Participants

At the time this draft was completed, the 802.2 Working Group   
had the following membership:

Roger Hislop, Chair

Oliver Holland, Vice Chair

Nilesh Khambekar, editor

Mike Cotton

Ken Baker

Gianfraco Miele

Gianni Cerro

Jerry Kalke

Apurva Mody

Participant8

Participant9

The following members of the 802.22.3 TG balloting committee voted on this 802.22.3 Standard. Balloters may have voted for approval, disapproval, or abstention.

[To be supplied by IEEE]

Balloter1

Balloter2

Balloter3

Balloter4

Balloter5

Balloter6

Balloter7

Balloter8

Balloter9

When the IEEE-SA Standards Board approved this 802.22.3 standard on <Date Approved>, it had the following membership:

[To be supplied by IEEE]

<Name>, Chair

<Name>, Vice Chair

<Name>, Past Chair

**Konstantinos Karachalios**, Secretary

SBMember1

SBMember2

SBMember3

SBMember4

SBMember5

SBMember6

SBMember7

SBMember8

SBMember9

\*Member Emeritus

Introduction

This introduction is not part of P802.22/D1, Draft 1 for Part22.3: Spectrum Characterization and Occupancy Sensing.

<Select this text and type or paste introduction text>

Contents

Table of Contents

[1. Overview 12](#_Toc494388672)

[1.1 Scope 12](#_Toc494388673)

[1.2 Purpose 13](#_Toc494388674)

[1.3 Application 13](#_Toc494388675)

[2. Normative references 13](#_Toc494388676)

[3. Definitions 14](#_Toc494388677)

[4. System Architecture 14](#_Toc494388678)

[4.1 System Roles 14](#_Toc494388679)

[4.2 System Block Diagram 14](#_Toc494388680)

[4.3 System Entities 15](#_Toc494388681)

[4.3.1 Entity Functions 16](#_Toc494388682)

[4.4 System Workflow 18](#_Toc494388683)

[4.5 System Entity Models 19](#_Toc494388684)

[4.5.1 Spectrum Sensing Device (SD) 19](#_Toc494388685)

[4.5.2 Task Manager (TM) 21](#_Toc494388686)

[4.5.3 Data Manager 24](#_Toc494388687)

[4.5.4 Data Client 26](#_Toc494388688)

[4.5.5 Data Consumer 27](#_Toc494388689)

[4.5.6 SD Proxy 27](#_Toc494388690)

[4.5.7 TM Proxy 28](#_Toc494388691)

[5. Interfaces, Messaging and Primitives 28](#_Toc494388692)

[5.1 SCOS Interfaces 29](#_Toc494388693)

[5.1.1 SCOS communication interfaces 29](#_Toc494388694)

[5.1.2 Data Client to TM Interface 30](#_Toc494388695)

[5.1.3 TM to SD Interface 30](#_Toc494388696)

[5.1.4 Data Manager to Data Consumer Interface 31](#_Toc494388697)

[5.2 SCOS Messaging 31](#_Toc494388698)

[5.2.1 Message Encoding 32](#_Toc494388699)

[5.2.2 Message Transport protocols 33](#_Toc494388700)

[5.2.3 Response Codes 33](#_Toc494388701)

[5.3 Primitives 33](#_Toc494388702)

[5.3.1 SD<>TM Messages 34](#_Toc494388703)

[5.3.2 DCli<>TM Messages 38](#_Toc494388704)

[5.3.3 SD<>DM Messages 43](#_Toc494388705)

[5.3.4 DCon<>DM Messages 47](#_Toc494388706)

[5.3.5 TM<>DM Messages 53](#_Toc494388707)

[6. System Administration and Security 57](#_Toc494388708)

[6.1 Administration 57](#_Toc494388709)

[6.1.1 SD Administration 58](#_Toc494388710)

[6.1.2 Sensing Task Administration 59](#_Toc494388711)

[6.1.3 SCOS Platform Behavior Specification 59](#_Toc494388712)

[6.2 Security Systems 60](#_Toc494388713)

[6.2.1 Scope 60](#_Toc494388714)

[6.2.2 Authentication, Confidentiality, and Integrity 60](#_Toc494388715)

[6.2.3 Authorization 61](#_Toc494388716)

[Annex A Informative: Reference Applications 62](#_Toc494388717)

[A.1 White Space device radio operation 62](#_Toc494388718)

[A.2 National spectrum regulation 62](#_Toc494388719)

[A.3 Research programs 63](#_Toc494388720)

[A.4 Law enforcement and public order 63](#_Toc494388721)

[A.5 Network Operator Applications 63](#_Toc494388722)

[Annex B Informative: Metadata Specification 64](#_Toc494388723)

[B.1 Metadata categorization 64](#_Toc494388724)

[B.1.1 System Metadata 64](#_Toc494388725)

[B.1.2 Current Status Metadata 65](#_Toc494388726)

[B.1.3 Sensing related Metadata 65](#_Toc494388727)

[B.2 Data Types 65](#_Toc494388728)

[B.2.1 Simple data type 65](#_Toc494388729)

[B.2.2 Complex data types 66](#_Toc494388730)

[B.3 Description of metadata 67](#_Toc494388731)

[B.3.1 Antenna 68](#_Toc494388732)

[B.3.2 Current Azimuth Beam Direction 69](#_Toc494388733)

[B.3.3 Current Elevation Beam Direction 69](#_Toc494388734)

[B.3.4 RF Front-End Metadata 69](#_Toc494388735)

[B.3.5 Calibration Metadata 70](#_Toc494388736)

[B.3.6 A.3.6 SDR Metadata 71](#_Toc494388737)

[B.3.7 Host Metadata 71](#_Toc494388738)

[B.3.8 Environmental Metadata 72](#_Toc494388739)

[B.3.9 Reference Geolocation 72](#_Toc494388740)

[B.3.10 Frequency 73](#_Toc494388741)

[B.3.11 Sample rate 73](#_Toc494388742)

[B.3.12 Bandwidth 73](#_Toc494388743)

[B.3.13 Timestamp 73](#_Toc494388744)

[B.3.14 Scanned Time 74](#_Toc494388745)

[B.3.15 Data Typology 74](#_Toc494388746)

[B.3.16 Sensing Technique 74](#_Toc494388747)

[B.3.17 Priority 75](#_Toc494388748)

[B.3.18 Timing 75](#_Toc494388749)

[B.3.19 Compression 75](#_Toc494388750)

[B.3.20 Format 75](#_Toc494388751)

[B.3.21 Sensing device name 76](#_Toc494388752)

[B.3.22 SCOS operator 76](#_Toc494388753)

[B.3.23 SD Mode 76](#_Toc494388754)

[B.3.24 Type of the sensing device 76](#_Toc494388755)

[B.3.25 Sensing Device ID 76](#_Toc494388756)

[B.3.26 SD certificate file 77](#_Toc494388757)

[B.3.27 SD Task Control metadata 77](#_Toc494388758)

[B.3.28 SCOS Association Metadata 77](#_Toc494388759)

[Annex C Normative Functional Requirements 79](#_Toc494388760)

[C.1 Data Client Requirement 79](#_Toc494388761)

[C.2 Data Quality and Definition 79](#_Toc494388762)

[C.3 Regulatory requirements 79](#_Toc494388763)

[C.4 Policy Management and Enforcement Requirements 79](#_Toc494388764)

[C.5 Sensor Location-Fixing Requirements 80](#_Toc494388765)

[C.6 Service Level Agreement Requirements 80](#_Toc494388766)

[C.7 Certification Requirements 80](#_Toc494388767)

[C.8 Technical Requirements 80](#_Toc494388768)

[C.8.1 Device classes and complexity 80](#_Toc494388769)

[C.8.2 Number of devices 81](#_Toc494388770)

[C.8.3 Real-time applications 81](#_Toc494388771)

[C.8.4 Channelization 81](#_Toc494388772)

[C.9 Security Requirements 81](#_Toc494388773)

[C.9.1 Intra-device Layer Security (physical interfaces) 82](#_Toc494388774)

[C.9.2 Inter-Layer Security 82](#_Toc494388775)

[C.9.3 Security of sensed data 82](#_Toc494388776)

[C.9.4 Security of analyzed or characterized spectrum data 83](#_Toc494388777)

[Annex D SCOS Operational Modes 84](#_Toc494388778)

[Annex E SCOS Topology Examples 87](#_Toc494388779)

[E.1 Non 802.22.3 compliant SDs and SCOS Cascading 87](#_Toc494388780)

[Annex F System Policy Model 89](#_Toc494388781)

[F.1 SCOS Policy 89](#_Toc494388782)

[F.1.1 The SCOS policy schema: 89](#_Toc494388783)

[F.1.2 Policy Evaluation 93](#_Toc494388784)

[F.1.3 SCOS Policy Examples 94](#_Toc494388785)

[Annex G Informative: Latency Requirements for Scans 97](#_Toc494388786)

[Annex H Informative: Regulatory Technical requirements 98](#_Toc494388787)

[Annex I Device and System Security Recommendations 99](#_Toc494388788)

[Annex J Radio performance requirements 100](#_Toc494388789)

[J.1 Sensitivity and Noise 100](#_Toc494388790)

[Annex K (informative) Sensing 101](#_Toc494388791)

[Annex L References 102](#_Toc494388792)

[Annex M (informative) Bibliography 104](#_Toc494388793)

Draft Standard for Information Technology – Telecommunications and information exchange between systems  
Wireless Regional Area Networks (WRAN)—Specific requirements  
Part 22.3: Spectrum Characterization and Occupancy Sensing

1. Overview
   1. Scope

The scope of this document is to define an open, global standard for building and inter-operating elements of a distributed radio signal detection and characterisation system. It establishes a high-level architecture, defines functional entities and their interfaces, specifies command & control messages and the parameters they would pass, and defines metadata describing the nature and configuration of the sensing system and the data it gathers. It also describes operating characteristics and behaviours of the components of the SCOS system, with reference implementations for common use cases. Further, it defines measurement parameters in such a way that they are of an equivalency with other related standards so that data gathered by an 802.22.3 compliant system can be consistently and meaningfully interpreted by users.

The intent of this standard is to allow interoperability between different 802.22.3 SCOS systems, or to allow elements of an 802.22.3-compliant SCOS system that are operated by different system owners to be interconnected. It would also allow proprietary and other sensing systems to be interconnected with the 802.22.3 system through proxy mechanisms.

The focus of this standard is on management and control of a sensor network, the ability for a variety of users to interrogate system capabilities and task scanning functions, the definition of a simple reference implementation for the sensing devices, and for the sensing data to be transmitted to an end-point.

* 1. Purpose

The purpose of the Spectrum Characterization and Occupancy Sensing (SCOS) system is to characterize and assess the occupancy of spectrum resource towards supporting its more efficient and effective use. The intent of the SCOS is to create a high-level architecture to support different spectrum sensing deployments, technologies and devices being shared to achieve economies of scale, and a broader availability and usage of sensing information from different sources. This will enable clients to acquire and use spectrum sensing information from a multiplicity of predefined independent systems to serve their goals.

* 1. Application

Various national regulators and government authorities are developing regulatory and policy frameworks to allow cooperative spectrum sharing approaches in order to optimize spectrum utilization. There is emphasis on greater spectrum efficiencies, spectrum sharing and spectrum utilization, which requires not only database-driven configuration of the radios, but systems that can provide spectrum occupancy at a particular location and at a particular time.

The Spectrum Occupancy Sensing (SCOS) System has many applications which include:

* On-demand spectrum survey and report
* Collaborative spectrum measurement and calibration
* Labelling of systems using the spectrum
* Spectrum planning
* Spectrum mapping
* Coverage analysis for wireless deployment
* Terrain and topology - shadowing and fading analysis
* Quantification of the available spectrum through spectrum observatories
* Complement database access for spectrum sharing by adding in-situ awareness and faster decision making
* Space-Time-Frequency spectrum hole identification and prediction where non-time-sensitive tasks can be performed at certain times and at certain locations, when spectrum use is sparse or non-existent
* Identification and geolocation of interference sources.

1. 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.

*[TO BE COMPLETED AS NECESSARY]*

[1] RFC-3339, "Date and Time on the Internet: Timestamps", Klyne, Newman, July 2002.

1. Definitions

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. [[2]](#footnote-2)

DCli Data Client

DCon Data Consumer

DM Data Manager

RF Radio Frequency

SCOS Spectrum Characterization and Occupancy Sensing

SM Sensor Manager

SD Sensing Device

TM Task Manager

1. System Architecture
   1. System Roles

The SCOS system identifies certain roles to meet the operational requirements and entities to meet the functional requirements. Roles are defined for individual/organizations and entities represent the software/hardware components in the system. The following roles have been identified based on operational requirements for the SCOS system.

**Sensor Owner:** The individual or organization that deploys and has administrative and physical control over the sensing devices (SD), where SDs are typically physical devices.

**Sensing Data Administrator:** The individual or organization that deploys and has administrative and physical control over the Data Consumers consisting of data stores or other consumers of spectrum sensing data delivered by the SCOS system.

**SCOS Administrator:** The individual or organization that deploys and has administrative and physical control over the SCOS System, consisting of the Task Manager and Data Manager.

**Data Client:** The individual or system that authenticates with the SCOS system through the Task Manager and causes a scan activity to be scheduled.

* 1. System Block Diagram

Figure 1: SCOS High Level System Block Diagram shows SCOS high level system block diagram. A SCOS Platform consists of sensor manager and sensor devices. A SCOS platform performs sensor task management and sensing data management. It provides APIs for scheduling sensing tasks and collecting the associated sensing data.

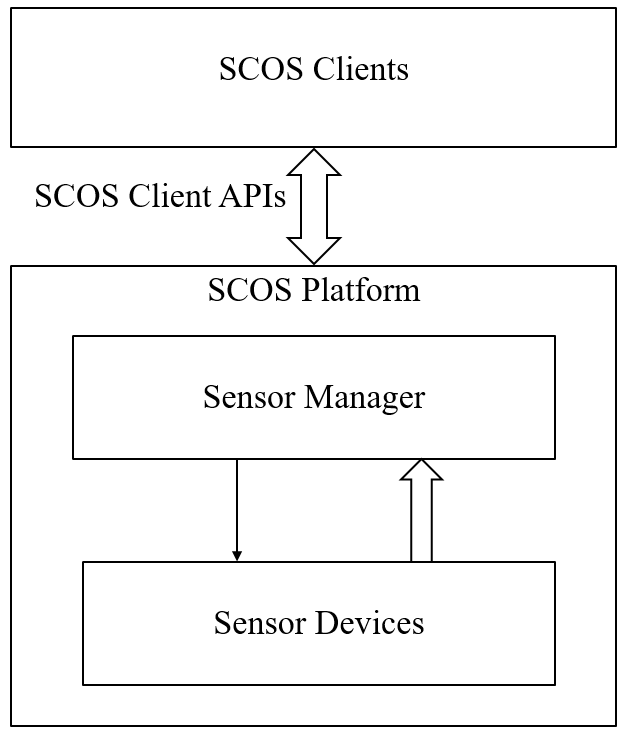


Figure 1: SCOS High Level System Block Diagram

Sensing task management forms the *SCOS control plane*. It involves processing sensing task requests from the SCOS clients and scheduling the sensing tasks to sensors.

*SCOS data plane* involves collecting sensor data, processing sensor data, and finally distributing it to the SCOS clients.

A SCOS system provides APIs for requesting sensing data, referred to as SCOS Data Request API, and consuming sensing data, referred to as SCOS Data Consumption API.

* 1. System Entities

The SCOS system architecture is defined based on the SCOS control plane and SCOS data plane. The part of SCOS client associated with the SCOS control plane is referred to as *Data Client* and the part associated with the SCOS data plane is referred to as *Data Consumer*. Data Client requests sensing data using SCOS Data Request API and Data Consumer consumes sensing data using SCOS Data Consumption API.

Sensor manager (SM) is split into Task manager (TM) and Data Manager (DM). Task manager represents the SCOS control plane functionality while Data manager is responsible for SCOS data plane functionality. Figure 2: SCOS Entity Diagram illustrates the SCOS architecture entities and API defined between the entities.

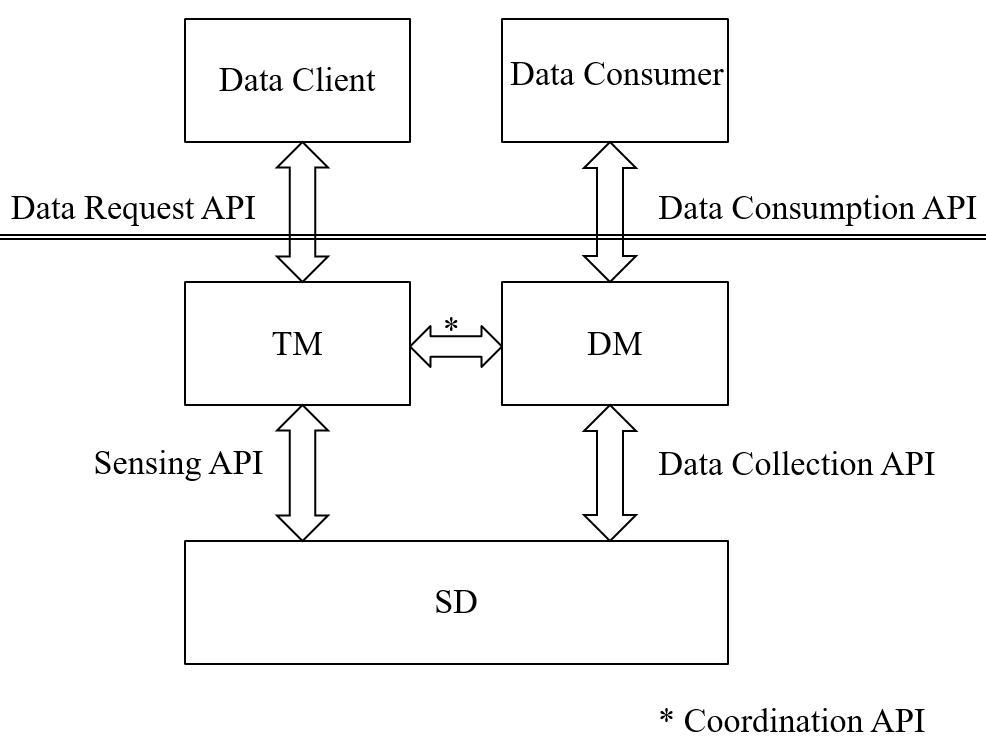


Figure 2: SCOS Entity Diagram

The TM handles sensing tasks from one or more Data Clients (DCli). The DM publishes sensing data to one or more Data Consumers (DCon). Thus, the topology mapping for sensing tasks is hence N:1:N for DCli:TM:SD. Similarly, the topology mapping for sensing data publishing is N:1:N for SD:DM:DCon.

The SCOS Platform provides a SCOS Request API to the Data Clients to initiate spectrum sensing tasks. The sensing tasks are scheduled by the SCOS platform on the sensing devices. The sensing devices send the sensing data to the SCOS platform. The SCOS platform publishes the sensing data to the Data Consumers using SCOS Data Distribution API.

The SCOS Platform provides Sensing API and Data Collection API to the Sensing devices for the purpose of associating sensor devices with the platform, performing sensing operations, and collecting the sensing data.

The SCOS control plane communication is synchronous; the data-plane communication is asynchronous.

It should be noted that within the SCOS system, the SDs shall not communicate with each other, or directly with the external roles using the SCOS system (DClis or DCons).

Entity Functions

**Data Client** is the entity that initiates a spectrum monitoring request to one or more Spectrum Task Managers (TM). Data Clients can be human or machine, and have various levels of privileges regarding what spectrum information collection can be initiated. Data Clients would determine where sensing data is to be transmitted, and authorization to access that data would rest with the owner of that data storage entity. and what spectrum information can be accessed from a Data Consumer.

A Data Client (user of the SCOS system) and TM (Task Manager) communicate by REST API to ask for available resources, and request a scan.

**Data Consumer** is a data store for storing spectrum information collected from the sensing network. There can be multiple DCs that sensing data is transmitted to by the Data Manager, and these can be, but not necessarily, associated with a specific Data Client.

The Data Manager transmits data to the DC via a Message Queue, and the Data Client interacts with the DC using their chosen mechanisms (out of scope of this standard)

**Spectrum Task Manager** (TM or Task Manager) manages a collection of Spectrum Sensing Devices (SD). Requests for spectrum measurements from Data Clients are inserted into a scan schedule on the TM for all its attached SDs, as far as possible under a set of slot availability rules. This schedule is synched to the appropriate SDs associated with the TM. Data from the SDs are collected at the Data Manager for transmission to one or more DCs for long term storage and processing.

The TM is associated with SDs (Sensing Devices) through a synchronous interface, where the TM enumerates and holds a list of available resources for each SD.

The TM stores and manages a schedule of scans against the sensing resources, and synchronizes this schedule with all SDs both on a change being made and periodically to ensure correct state.

**Data Manager** receives transmissions of packaged scan data from SDs, and retransmits it to one or more destinations, as defined by the policies associated with each Data Client (the source of scan requests)

The “Data Manager” applies any policies and then handles the Store & Forward to one or more DCs using a Message Queue or Streaming Mechanism

**Task Manager and Data Manager together form the SCOS Manager**, and each can be on the same platform or separate platforms.

**Spectrum Sensing Device (SD)** is the sensing hardware that collects the spectrum data requested by the TM on behalf of each Data Client. The SDs may exist with various levels of sophistication. The less sophisticated might be capable of measuring only one band, at only one resolution with little on-board processing. Other sensors may incorporate sophisticated antenna techniques, multiple bands, calibration processes, on-board data processing and/or storage and/or be capable of mobile operation.

An SD performs the scans in the schedule, and transmits the data and associated metadata through an asynchronous interface (message queue, or real-time stream) to a “Data Manager” that performs system data validation (i.e. that a transmission is received completely, partial scans are consolidated, etc).

**SD Proxy** enables an TM talk to any other proprietary sensing hardware, acting as a software translation mechanism that translates between commands/metrics/etc. It would need to be custom written for the particular device it talks to.

**SM Proxy** facilitates an SM talk to SM of another SCOS platform, with the downstream SM appearing as if it were an SD with a set of resources it provides. This downstream SCOS system would need to be 802.22.3 compliant.

The flow of instructions and data is described in Figure 3: SCOS Functional Block Diagram.

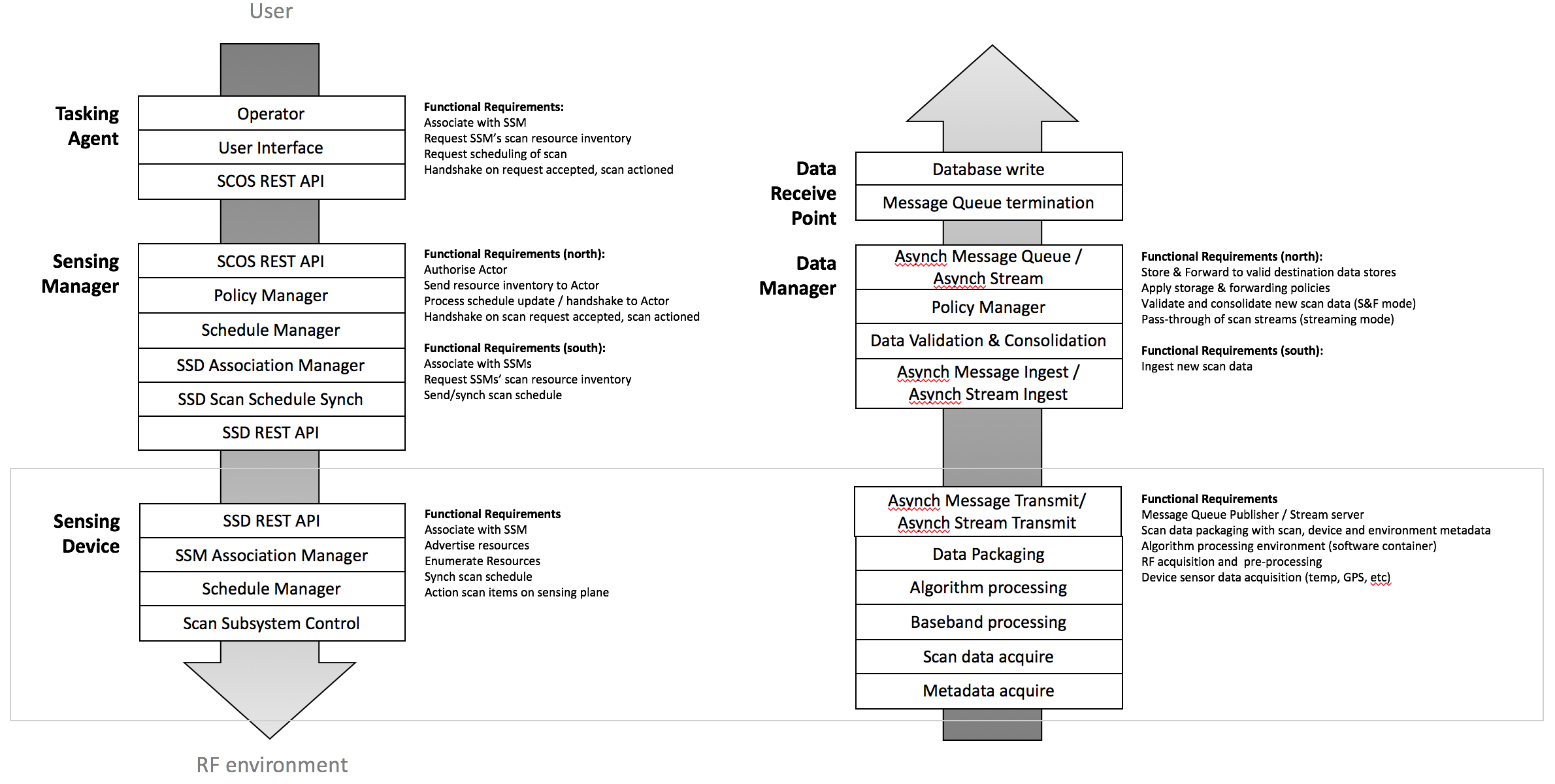


Figure 3: SCOS Functional Block Diagram

* 1. System Workflow

The Data Clients interact with Task Manager through the 802.22.3 defined Data Request API. The tasking API facilitates querying the available sensing resources and schedule sensing tasks as permitted by resource availability, authorization level and system policies.

The sensing devices are associated with the SCOS system using the SCOS sensing API. The TM maintains an inventory of the sensing resources along with the SD capabilities and parameters described according to this 802.22.3 standard.

Following is the typical workflow within the SCOS system.

* The Data Client submits a scanning task into a schedule managed by the TM using the SCOS Data Request API defined in this 802.22.3 standard
* The TM maintains the schedule of tasks and synchronises this schedule of tasks to the SD using the SCOS sensing API defined in this 802.22.3 standard
* Within the SD, radio energy shall be collected by an antenna and transferred through an interconnect to a signal conditioner. Conditioned signal will be then transferred to a signal processing system to produce a baseband signal that can be quantised and passed in digital form to be analysed through whichever sensing technique is offered by the SD
* The data from this analysis, and the associated metadata that includes the sensing technique and environment, hardware, software and configuration parameters as defined in this standard, can be temporarily stored on the local device before it shall be transmission to an end point.
* SD transmits the sensing data to the Data Manager using the SCOS Data Collection API.
* The DM establishes the endpoint of data package transmissions according to the Data Client’s nominated DCs, and in accordance with the policies defined. DM publishes the data to the DCs.
* TM and DM coordinate the task status.
* Finally, TM reports the success or otherwise for the sensing task back to the original Data Client.
  1. System Entity Models

Spectrum Sensing Device (SD)

SDs convert radiative electromagnetic energy into a voltage, which is then sampled. The samples can then be processed in various ways to provide information on the immediate RF environment, e.g., amplitude statistics versus frequency, amplitude and phase versus time at a given frequency, occupancy statistics, angle of arrival.

* + - * 1. Hardware Model

A simplified hardware block diagram of a general SD model is depicted in Figure 4: SD Simplified Hardware Model Block Diagram. SD hardware designs are not required to have each component shown in the block diagram. Specifics for each component (e.g., presence, model, operational parameters), however, is required metadata when SD capabilities are queried by the TM. This SD definition metadata is also accompanied with the output data messages.

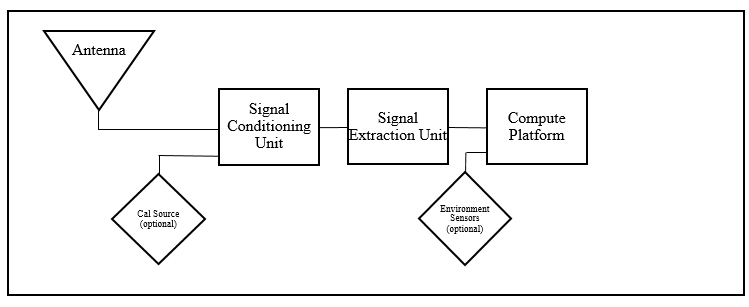


Figure 4: SD Simplified Hardware Model Block Diagram

The SD is composed of the following functional elements, as follows:

● Functional element 1 – Antenna: An antenna used to collect RF energy. This is fed to functional element 2 over a hardware interface (interconnect cable)

● Functional element 2 – Signal Conditioning Unit: An RF front end unit consisting of (all or some of) an RF switch (optional, with the ability to accept an optional calibration signal), filter, Low Noise Amplifier, mixer. This sends the conditioned signal to functional element 3 over an analogue hardware interface (interconnect cable/track).

● Functional element 3 – Signal Extraction Unit: Analogue Digital Converter, spectrum analyzer or Software Defined Radio to act as a baseband processor, performing a demodulation of the conditioned signal and acquires the baseband signal. This sends a digitized signal over a digital interface (interconnect)

● Functional element 4 – Compute Platform: that provides

o A signal processing function with a signal detection and/or classification algorithm. It sends detection/classification data to metadata consolidation and packaging function over a software interface.

o A metadata consolidation and data packaging function that combines sensing data with environmental inputs (where implemented), hardware, operating and system-configured metadata. It sends data packages to the transmission system over a software interface.

o A transmission unit that transmits scan data to the destination system over a best-effort IP connection.

The Compute Platform sends necessary command and control signals to Functional element 2 (Conditioning Unit) and Functional element 3 (Extraction Unit). It receives data from the Sensor/SDR, and polls any environment sensor input devices for necessary metadata items, such as GPS location. Interaction of the various elements is described in Figure 5: SD Functional Elements.

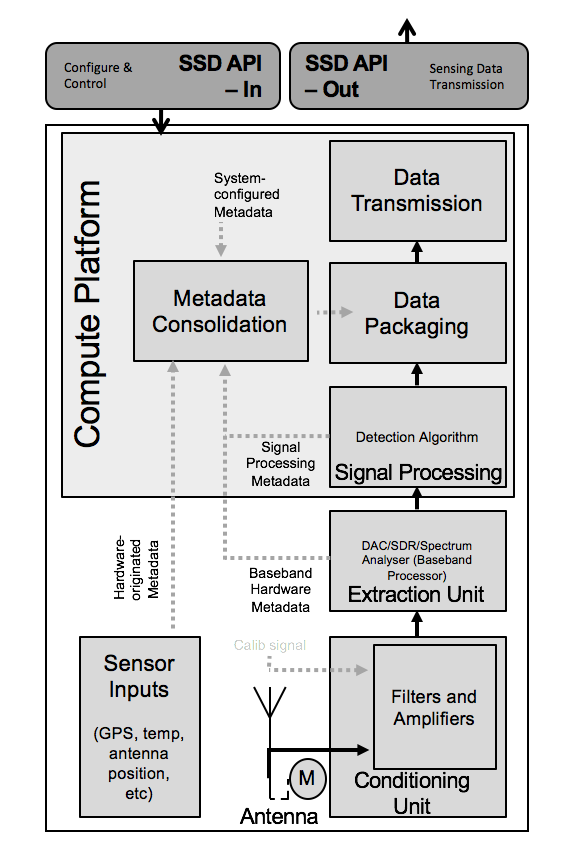


Figure 5: SD Functional Elements

This block diagram can be split into the hardware layer and the software processes that run alongside. These hardware blocks or software services generate metadata that is associated with each item.

* + - * 1. SD Calibration Model

SD calibration can be done in the lab at build/commissioning time, and stored as a calibration file on the SD. Further, an SD with a self-calibration capability can be instructed through an administrative interface (not Data Client request) to perform a calibration using a local calibration source.

* + - * 1. SD Algorithm Model

The algorithm model is described in terms of

● inputs into black box: the identity of the USER and TM requesting the scan, the measurement parameters, which algorithm is to be used; and

● outputs from the black box: the identity of the USER and TM requesting the scan, the requested scan parameters, the identification of the algorithm model, and the processed results.

The following table enumerates the parameter definition object for the Algorithm Model.

Table 1 SD Algorithm Model Description

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: AlgorithmSet  DATA TYPE: Array of String | Required | Names of algorithms supported by the SD.  The maximum length of the ID string is 64 octets. |

The following algorithms can be specified. At least once algorithm model needs to be supported by SD. Support for GenericEnergyDetection is normative. The standard allows development of advanced algorithms, which shall be identified in the table.

Table 2 SD Detection Algorithm Options

|  |  |
| --- | --- |
| Scan Algorithm | Description |
| GenericEnergyDetection | Normative. |
| CyclicFeatureDetection | Optional |
| CustomScanAlgorithm | Optional |

It is the responsibility of the SCOS Administrator to publish algorithm definitions externally. The implementation does not need to be publicly accessible. Where these algorithms are non-proprietary, the metadata should include explicit links to repositories providing algorithm details, sample code and definition, on open access sites such as GitHub.

Annex B defines how control plane and data plane information specification. In Annex B, different categories of metadata are identified and described.

Task Manager (TM)

The TM is primarily responsible for sensing task management. Sensing task management involves

* Scheduling of Scan based on the requests from the Data Clients
* Enforcing policy to meet the regulatory and administrative requirements
* Communication with SDs, DCs, and DM
  + Data Request Interface with Data Clients
  + Sensing Interface with SDs
  + Sensing Coordination with DM
    - * 1. TM Task Scheduling

SCOS platform provides sensing as a service. SCOS facilitates to make sensing requests in terms of frequency, start time, end time, scan duration, repetition. SCOS data request API provides the ability to specify additional sensing options..

A Data Client may indicate whether the desired scan slots are “Exact Time” slots or “Nearest Time” slots. The scheduler on the TM uses this indication to try meet the DCli request and confirms whether scan service request is accepted or refused.

The following table enumerates the parameter definition object for scanTask.

Table 3 Sensing Scan Task Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: TaskID  DATA TYPE: String | Required | Unique ID for the Spectrum Scan.  The maximum length of the ID string is 64 octets. |
| NAME: TaskDuration  DATA TYPE: number | Required | Duration of scan in milliseconds. |
| NAME: TaskStartTime  DATA TYPE: Time | Required | The start time for the task. |
| NAME: TaskRepeatInterval  DATA TYPE: Number | Optional | The interval in seconds after which the task needs to be repeated. |
| NAME: TaskRepeatCount  DATA TYPE: Number | Optional | The number of times the task needs to be repeated.  The maximum length of the ID string is 64 octets. |
| NAME: TaskEndTime  DATA TYPE: Time | Conditional | The end time for the task. If repeatInterval and repeatCount are specified, TaskEndTime is not required. |
| NAME: TaskAttributes  DATA TYPE: Integer | Optional | Currently following task attributes can be specified.  0 = Exact time.  1 = Nearest time. |
| NAME: TaskOptions  DATA TYPE: Integer | Optional | Custom options. |

* + - * 1. TM Simplified Model

Figure 6: TM Functional Block Diagram shows a simplified model for TM.

● Sensor Management: TM maintains the information from sensors obtained during SD association message exchange. The sensor information is identified in Annex B. This information is used by the TM during task scheduling for determining which set of sensors could be used for performing the scan.

● Sensing Task Management: TM maintains information about to-be-scheduled, scheduled, and on-going tasks. The information about to-be-scheduled tasks is used in task scheduling, the information about scheduled tasks is synchronized with the SDs. The information about the ongoing tasks is used in task query/coordination/notification with TAs, DM, and SDs.

● Policy Enforcement: A key part of SCOS administration is enforcing policies for spectrum sensing. The TM ensures that the DCli issuing a scan request is authorized to perform the requested scan, the scanning parameters comply with the regulatory policy for the location, frequency, time, and resolution

● Task scheduling: Task scheduling involves enforcing policy for the requested scan, identifying a set of eligible and available sensors using the sensor information, associating chosen sensors to the task, assigning scanTaskID to the task, updating the task status in the task information.

● Sensing coordination: The TM needs to coordinate the status of tasks with DM for example, TM assigns a DM for each of the scheduledTasks. The scanTaskID assigned by the TM is used by SDs when SDs provide sensing data to the DM.

● SCOS Data Request API: The TM communicates with the TAs using the Data Request API, provides response to the methods under the Data Request APIs, provides notification to the TAs upon task completion or error events.

● SCOS Sensing API: The TM communicates with SDs using the Sensing API, receives the sensor capabilities, and provides a spectrum scan schedule to the sensors along with necessary information for sending sensing data to the DM.

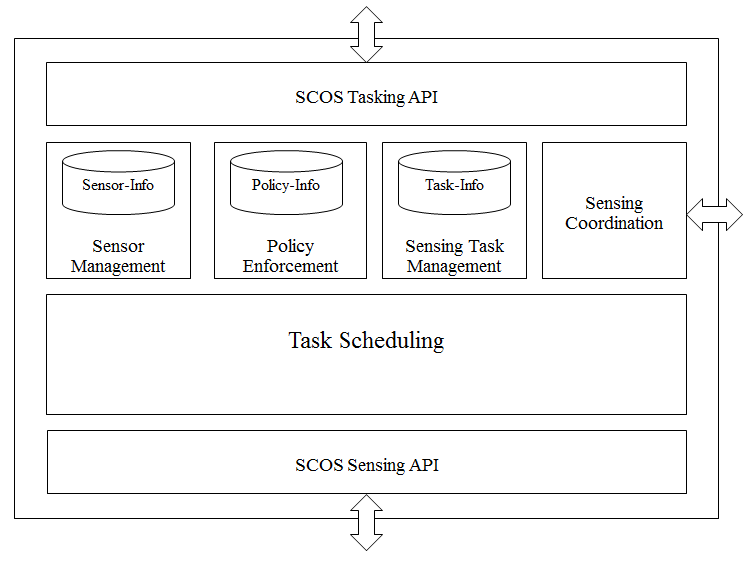


Figure 6: TM Functional Block Diagram

* + - * 1. SCOS Association Metadata

The following table enumerates sensing manager parameters toward associating with SCOS.

Table 4 TM Parameter Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: TMID  DATA TYPE: String | Required | Unique ID for the Task Manager.  The maximum length of the ID string is 64 octets. |
| NAME: TMSID  DATA TYPE: String | Required | Unique ID for the SMS.  The maximum length of the ID string is 64 octets. |
| NAME: SCOSOperator  DATA TYPE: String | Required | The registered name of the SCOS operator.  The maximum length of the ID string is 64 octets. |
| NAME: TMURL  DATA TYPE: String | Required | The URL for reaching to the TM.  The maximum length of the ID string is 256 octets. |
| NAME: TMCertFile  DATA TYPE: String | Required | The path of the TM certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: TMKeyFile  DATA TYPE: String | Required | The name of the TM certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: TMCAFile  DATA TYPE: String | Required | The name of the trusted certificate authority.  The maximum length of the ID string is 256 octets. |

Data Manager

The DM is primarily responsible for sensing data management. The standard identifies following key functionalities within the DM entity

* Collecting spectrum scan data from SDs
* Distributing sensing data to the DCons
* Policy enforcement to meet the regulatory and administrative requirements.
* Communication with SDs, DCs, and TM
  + Data collection interface with SDs
  + Data distribution to the SDs
  + Sensing task coordination with the TM
    - * 1. Sensing Data Management

The SDs send spectrum measurements as requested by the scheduled scan. B.3.27.2 describes the sensing data. The sensing data is identified by ScanTaskId, SDID, and timestamp. The SD also provides envInfo which includes environmental data including GPS, humidity, and temperature. The DM validates the received data, consolidates the data with other received data for the scanning task, stores it internally for further distribution to DCs, and provides acknowledgement to the receives sensing data to the SDs.

Table 5 Scan Measurement Data Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: dataFormat DATA TYPE: Integer | Required | The format of the output data as specified in B.3.27.2. |
| NAME: sizeData  DATA TYPE: Integer | Required | The number of measurements. |
| NAME: measData  DATA TYPE: Array of Complex | Required | The complex measurement values. The size of the array is defined by sizeData. |

* + - * 1. DM Simplified Model

Figure 7: DM Functional Block Diagram shows a simplified model for DM.

* Sensing data management: DM maintains data for the on-going tasks. The data received from the SDs needs to be stored internally for distribution to DCs. The DM may also need to have the capability to hold data for certain privileged tasks for a short duration identified by policy for the sensing data.
* Sensing data distribution: DM maintains the information regarding DCs and their associated scanning tasks. Sensing data distribution involves providing necessary reliability depending on the chosen transport.
* Policy enforcement: A key part of SCOS administration is enforcing policies on the sensing data. The DM ensures that the DCs issuing a subscription request for the sensing data are authorized to receive the data.
* Data validation and consolidation: The DM validates the data received from the SDs against the specified details from the task such as location, frequency, time, and measured data format. It consolidates data based on scanning task requirements.
* Sensing coordination: The DM needs to coordinate the status of tasks with TM for example, the expected sensing data from specific SDs for specific sensing tasks; also, availability of DM resources for future scanning tasks which could be used TM in choosing the DM.
* SCOS Sensing Data Collection API: The DM communicates with the SDs using the Data Collection API. It receives data for specific sensing tasks from specified SDs as coordinated with the TM.
* SCOS Sensing Data Distribution API: The DM communicates with DCs, implements the API for distributing the sensing data to the authorized DCs.

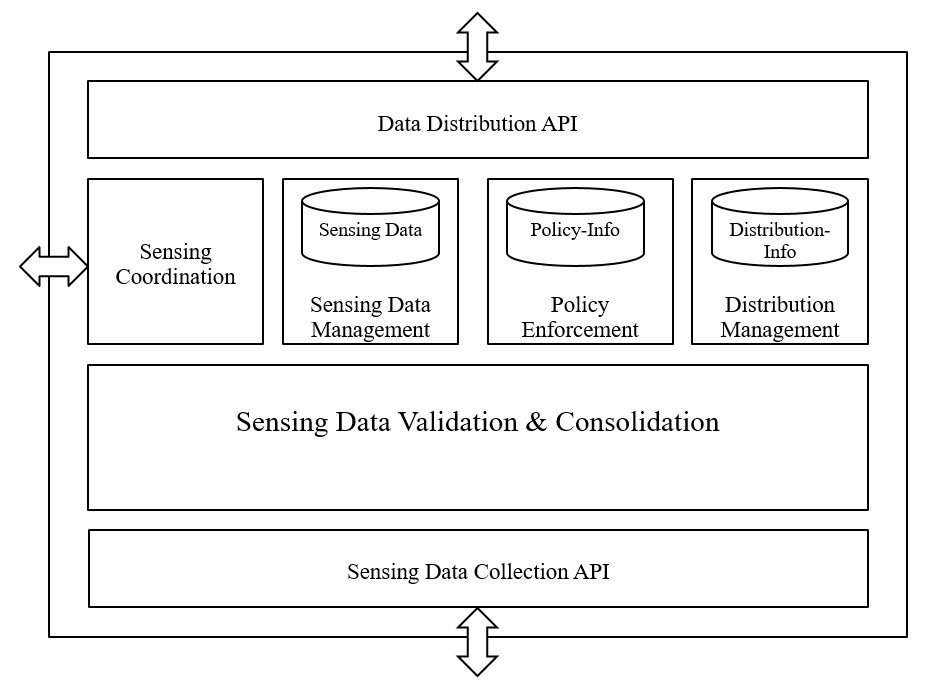


Figure 7: DM Functional Block Diagram

* + - * 1. DM Metadata

The following table enumerates the parameter definition object for Data Manager.

Table 6 Data Manager Parameter Definition Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DMID  DATA TYPE: String | Required | Unique ID for the Data Manager.  The maximum length of the ID string is 64 octets. |
| NAME: SMSID  DATA TYPE: String | Required | Unique ID for the SMS.  The maximum length of the ID string is 64 octets. |
| NAME: SCOSOperator  DATA TYPE: String | Required | The registered name of the SCOS operator.  The maximum length of the ID string is 64 octets. |
| NAME: DMURL  DATA TYPE: String | Required | The URL for reaching to the TM.  The maximum length of the ID string is 256 octets. |
| NAME: DMCertFile  DATA TYPE: String | Required | The path of the TM certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DMKeyFile  DATA TYPE: String | Required | The name of the TM certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DMCAFile  DATA TYPE: String | Required | The name of the trusted certificate authority.  The maximum length of the ID string is 256 octets. |

Data Client

The standard provides SCOS Data Request API to the Data Clients for performing spectrum scans. Using the API:

* DClis are associated with the TM of the SCOS platform
* DClis perform resource discovery
* DClis request spectrum scans at the desired time, frequency, and locations
* DClis specify the Data consumers that are authorized to receive the sensing data for the scans.
* DClis optionally perform query about the sensing tasks.

Data Clients communicate the information provided by the TM to the SCOS platform user. Data Clients may implement certain logic/policy to automate the spectrum sensing process.

* + - * 1. Data Client Metadata

The following table enumerates the parameter definition object for Data Client.

Table 7 Data Client Parameter Definition Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliName DATA TYPE: string | Required | The name of the DCli registered with the SCOS operator.  The maximum length is 64 octets. |
| NAME: DCliID  DATA TYPE: String | Required | Unique ID for the DCli.  The maximum length of the ID string is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: DCliCertFile  DATA TYPE: String | Required | The path of the DCli certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DCliKeyFile  DATA TYPE: String | Required | The name of the DCli certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DCliCAFile  DATA TYPE: String | Required | The name of the trusted certificate authority.  The maximum length of the ID string is 256 octets. |

Data Consumer

The standard provides SCOS Data Distribution API to the DCons for requesting the spectrum sensing data. Using the API,

* DCons are associated with the DM of the SCOS platform
* DCons request sensing data for specific sensing tasks
* DCons receive the data streamed or forwarded by the DM

The DCons would typically ingest and store the received data as defined the SCOS user specific logic/policy.

* + - * 1. DCon Metadata

The following table enumerates the parameter definition object for a data consumer.

Table 8 Data Consumer Parameter Definition Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DConName DATA TYPE: string | Required | The name of the DCon registered with the SCOS operator.  The maximum length is 64 octets. |
| NAME: DConID  DATA TYPE: String | Required | Unique ID for the DCon.  The maximum length of the ID string is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: DConCertFile  DATA TYPE: String | Required | The path of the DCon certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DConKeyFile  DATA TYPE: String | Required | The name of the DCon certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DConCAFile  DATA TYPE: String | Required | The name of the trusted certificate authority.  The maximum length of the ID string is 256 octets. |

SD Proxy

An SD Proxy interfaces with SCOS on the behalf of proprietary SD. SD proxy interaction to SCOS is identical to that of the SD except that while registering it registers with SDType as SDProxy. Following diagram illustrates the SD-Proxy’s functional architecture.

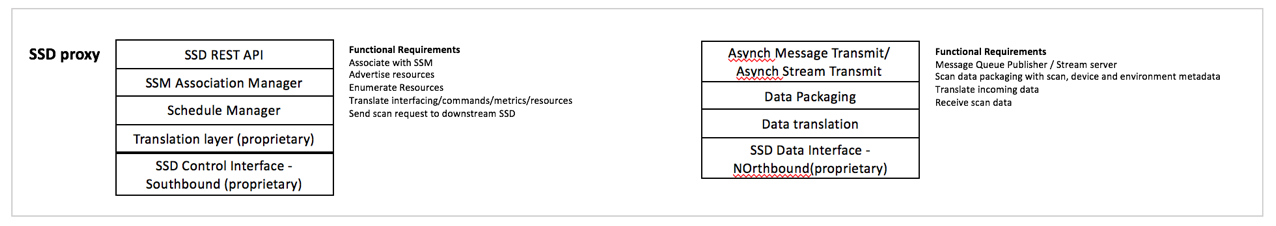


Figure 8: SD Proxy Functional Block Diagram

TM Proxy

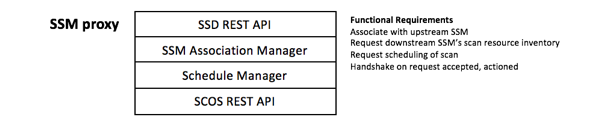
****

Figure 9 TM Proxy Functional Block Diagram

The TM Proxy enables cascading of the SCOS system. There are two options for cascading

* TM Proxy acts as Proxy for all the underlying SDs and registers them with the SCOS system. Thus, all interactions with the actual underlying SDs are mediated by the TM-Proxy.
* TM-Proxy uses TM-TM interface and the two TMs interoperate with each other. This mode is planned to be introduced in the later version of the standard.

1. Interfaces, Messaging and Primitives

Figure 10: Simplified Interactions Model illustrates a simplified SCOS interactions model. Here, we re-iterate a few architectural details.

* A SCOS platform is composed of sensors and a sensor manager. SCOS system provides sensing as a service.
* The SCOS system control plane is responsible for sensing task management. It enables the clients to SCOS platform to make requests for sensing data. The SCOS system data plane collects the sensing data and enables the SCOS platform clients to consume the requested data. In this regard, the sensor manager is split into Task Manager and Data Manager. The SCOS control plane counterpart of a SCOS platform client making request to sensing data is Data Client. The SCOS data plane counterpart of a SCOS platform client consuming the sensing data is Data Consumer.
* A Data Client connects to Task Manager using well-known address of the Task Manager and *authenticates* itself with the Task Manager. Upon successful authentication, the Dcli performs query on SCOS platform using *Resource Discovery* mechanism. A DCli can the *request sensing data* from the SCOS platform. TM provides *notification* of the completed sensing tasks to the DCli. A Dcli may *request status* on the requested sensing data.
* An SD may actively connect to the sensor manager on the SCOS platform or may passively wait to be contacted by a sensor manager. The SD and TM perform mutual *authentication*. Upon successful authentication, the TM performs *capability discovery* on the SD. The TM *schedules sensing tasks* on the SDs. The TM maintains periodic heartbeat with the SDs.
* The SDs connect with the DM that handles the SCOS data plane functions. After successful mutual authentication, SD pushes the sensing data to the DM.
* The Data Consumers connect with the DM in order to consume the sensing data associated with the sensing request from its counterpart DCli. DCon authenticates itself with the DM. Upon successful authentication, the DCon can receive the sensing data published by the DM.

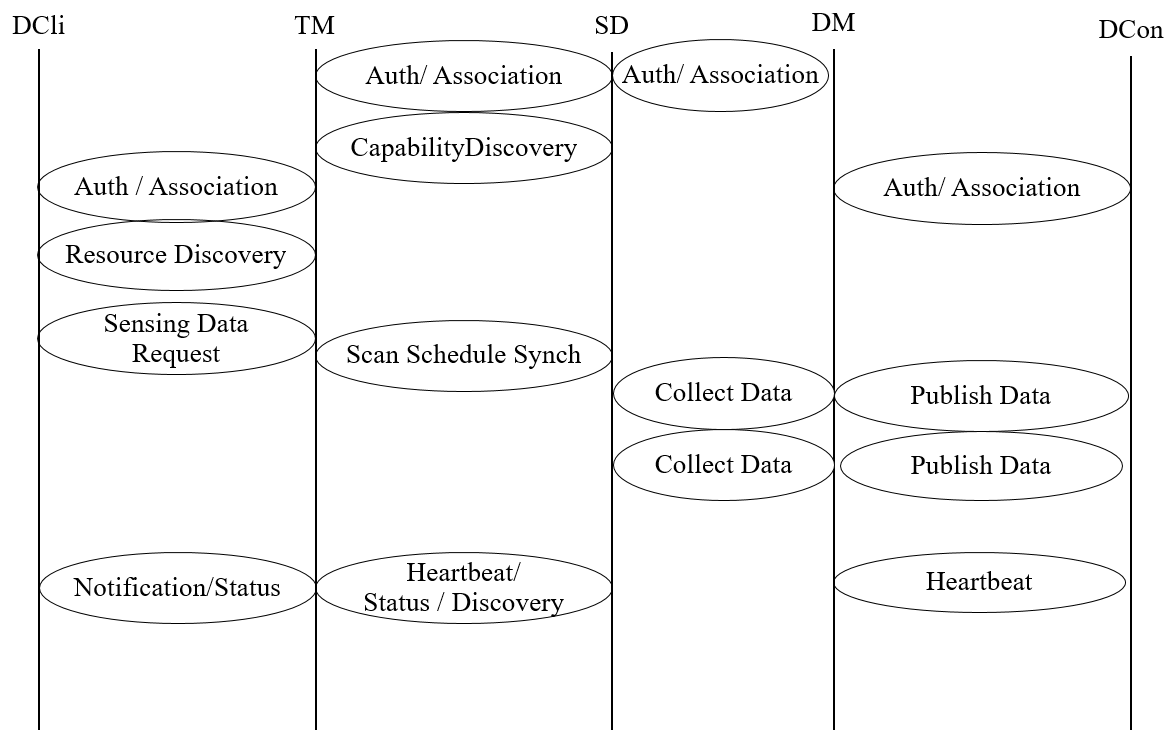


Figure 10: Simplified Interactions Model

* 1. SCOS Interfaces

SCOS communication interfaces

The following are the key SCOS communication interfaces.

* DCli and TM Interface
  + The interface between DCli and TM is asynchronous.
  + The interactions on this interface are specified in the SCOS Data Request API.
* TM and SD Interface
  + The interface between TM and SD is required to be synchronous.
  + The interactions on this interface are specified in the SCOS Sensing API.
* SD and DM Interface
  + The interface between SD and DM is asynchronous.
  + The interactions on this interface are specified in the SCOS Data Collection API.
* DM and DCon Interface
  + The interface between SD and DM is required to be asynchronous.
  + The interactions on this interface are specified in the SCOS Data Consumer API.

Data Client to TM Interface

* + - * 1. Authentication and Registration

These procedures define the association and authentication process for an TM and Data Client entity to connect and communicate. They include facilities to prevent spoofing based on shared key exchange. Once an TM is authenticated and registered to a Data Client, the Data Client can then discover the capabilities of the TM and its associated SD’s. The Data Client may then define and make sensing requests to the TM, which include a designation of the Data Consumer(s) to which the data is to be sent. The TM will notify the Data Client when measurements are successfully completed (or not) and available at the Data Consumer.

* + - * 1. Resource Discovery

Resource Discovery is the process of informing the Data Client of what capabilities that the TM has with regard to what types of measurements, what bands can be measured and associated measurement parameters that can be specified and controlled and over what locations.

This takes the form of a resource/capability message object and the current scan schedule per SD.

* + - * 1. Scan Request

The Scan Request message from the Data Client to the TM includes the parameters of the desired spectrum measurement to be made and any associated processing to be performed by either the SD or the TM. This scan request is wrapped in a scheduling task description, defining the time the scan is to be made, the repetition rate (if applicable), the locations, etc. When the scan parameters in their scheduling wrapper are received by the TM it will be validated as possible to be executed (i.e. the resources requested meet the SSMs schedule of resources available), and either acknowledged as being queue, or a refusal is returned to the Data Client. If a scan schedule is updated for a particular SD, it is then replicated down to that SD.

TM to SD Interface

* + - * 1. Authentication and Registration

These procedures define the association and authentication process for an SD and TM entity to connect and communicate. They include facilities to prevent spoofing based on shared key exchange. Once an SD is authenticated and registered to a TM, the TM can then discover the capabilities of the SD. An TM will have associated with it at least one SD. The TM may then assign sensing requests to the appropriate set of SDs in order to fulfil the sensing request of the Data Client.

* + - * 1. Status and Discovery

The Status and Discovery process serves two functions. The first is to inform the TM of what capabilities that the SD has with regard to what types of measurements, what bands can be measured and associated measurement facilities (such calibration, antenna control, mobility, storage, processing) that can be specified and controlled and over what locations. The SD will transmit a package describing its capabilities and available resources at time of authentication/discovery, and if there is any change in its configuration. The second function is to maintain association with the TM. It will transmit a heartbeat periodically to indicate it is still associated with the TM. If it is to disconnect, it will transmit a disassociation message (e.g. if it is rebooting or about to go into an offline mode).

* + - * 1. Scan Request

The Scan Request message originating from the TM is sent to the appropriate SDs for execution as a scan schedule. It includes the parameters of the desired spectrum measurement to be made based on knowledge of the SD’s capabilities. This request will include the time to make the measurement, the repetition rate (if applicable), the locations, etc. and the format of the measured data. In the case of a single, once-off scan, the schedule will indicate no repetition.

Message Parameters are captured in Table 3.

Data Manager to Data Consumer Interface

* + - * 1. Authentication and Registration

These procedures define the association and authentication process for a Data Consumer and DM entity to connect and communicate. They include facilities to prevent spoofing based on shared key exchange. Once a Data Consumer is authenticated and registered with a DM, the DM is then authorized to cause data to be delivered to the Data Consumer based on the privileges of the Data Consumer and the DM. The Data Consumers can be grouped into Data Consumer Groups, where a transmission of data from the DM is delivered to multiple Data Consumers.

* + - * 1. Data Manager

These procedures define and enable the storage of data from the DM to the Data Consumer. The successful reception of this data initiates a notification of the initiating Data Client that requested that data.

* 1. SCOS Messaging

The communication between each of the entities defined above can be grouped and defined within the Interface Categories shown in Figure 11: SCOS Message Sequence.

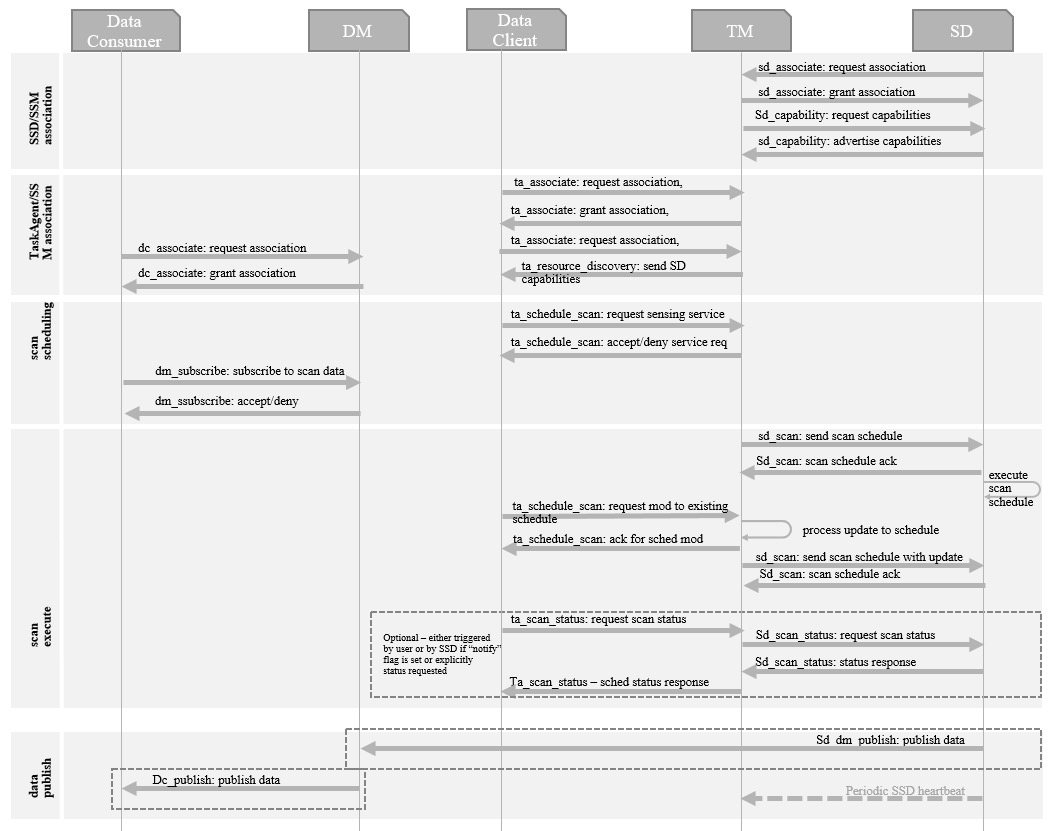


Figure 11: SCOS Message Sequence

Message Encoding

SCOS messages are encoded in JavaScript Object Notation (JSON). JSON is a language-independent data-interchange format that is easy for humans to read and write. There are code and functions readily available in C, C++, C#, Java, JavaScript, MATLAB, Perl, and Python for parsing and generating JSON. It is a lightweight alternative to XML, commonly used to transmit data between server and browser application.

The first five fields are the same for all messages; they are:

1. Ver = Schema/data transfer version with the major.minor.revision syntax (string)

2. Type = Type of JSON message (string) {“Sys”, ”Loc”, or “Data”}

3. SensorID = Unique identifier of sensor (string of URL unreserved characters)

4. SensorKey = Authentication key given out by MSOD (integer)

5. t = Time [seconds since Jan 1, 1970 UTC] (long integer)

Each message begins with a header comprised of attribute-value pairs in ASCII characters.

If an attribute is not relevant to the sensor implementation, then the value is set to NaN or "NaN".

The following are specific formatting rules to be followed:

* All timestamps, i.e., t (defined above)and t1 (to be defined in Data message description) will be reported as seconds since 1/1/1970 midnight UTC in the UTC time zone.
* String values must only contain URL unreserved characters (i.e., uppercase and lowercase letters, decimal digits, hyphen, period, underscore, and tilde), and
* Field names cannot start with an underscore because that convention is reserved for internal implementation-specific uses.

Message Transport protocols

The SCOS standard is transport-agnostic. The standard defines requirements for the transport protocol. The implementers may choose appropriate transport protocol that meets these requirements and suits to the use-case.

Response Codes

0: success

100 – 199: error events related to the entity association and disassociation

200 – 299: error events related to the entity configuration and policy enforcement

300 – 399: error events related to the scanning procedures

400 – 499: error events related to the data dissemination procedures

500 – 599: error events related to the heartbeat procedure and SCOS entity/platform health

600 – 699: error events related to the SCOS infrastructure administrative procedures.

* 1. Primitives

Each message (in general) will begin with a header as shown in the following table.

Table 9 SCOS Messager Header Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: version  DATA TYPE: String | Required | IEEE 802.22.3 SCOS protocol version.  The maximum length is 64 octets. |
| NAME: scosmode  DATA TYPE: Integer | Required | The mode for SCOS system. |
| NAME: scosmethod DATA TYPE: String | Required | The SCOS method in the context of the communication. The scaos methods are listed in the message descriptions.  The maximum length is 64 octets. |
| NAME: msgtype DATA TYPE: Integer | Required | The valid message types are request and response. (1=Request, 2=Response 3= Notification 4=AdminCmd 5=AdminCmdResponse) |
| NAME: timestamp  DATA TYPE: Time | Required | Timestamp associated with the communication.. |

The following are specific formatting rules to be followed to avoid problems when messages are ingested into MSOD: (1) All timestamps, i.e., t (defined above)and t1 (to be defined in Data message description) will be reported as seconds since 1/1/1970 midnight UTC in the UTC time zone. (2) String values must only contain URL unreserved characters (i.e., uppercase and lowercase letters, decimal digits, hyphen, period, underscore, and tilde), and (3) Field names cannot start with an underscore because that convention is reserved for MSOD internal use.

*The data fields in the JSON message descriptions below are required fields. If an attribute is not relevant to the sensor implementation, then the value is set to NaN or "NaN".*

SD<>TM Messages

Table 10 enumerate the messages exchanged between an SD and the TM.

Table 10 SCOS Messages between SD and TM

|  |  |  |
| --- | --- | --- |
| scos\_method\_name | JSON Array Name of Request Message | JSON Array Name of Response Message |
| “sd\_associate” | *sdAssociateRequest* | *sdAssociateResponse* |
| “sd\_capability” | *sdCapabilityRequest* | *sdCapabilityResponse* |
| “sd\_scan” | *sdScanRequest* | *sdScanResponse* |
| “sd\_heartbeat” | *sdHeartbeatRequest* | *sdHeartbeatResponse* |
| “sd\_disassociate” | *sdDisassociateRequest* | *sdDisassociateResponse* |

* + - * 1. SD-TM Association Message Exchange

Table 11 describes the sdAssociateRequest JSON object.

Table 11 SD Association Request Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDName DATA TYPE: string | Required | The name of the sensing device registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: SDMode DATA TYPE: Integer | Required | The mode in which sensing device operates. (1=online, 2=offline) |
| NAME: SDType DATA TYPE: Integer | Required | The type of the sensing device. (1=SDFull, 2=SDProxy) |
| NAME: SDID DATA TYPE: string | Conditional | The unique ID assigned to the sensing device. If ID is not pre-assigned, this is left empty. The maximum length is 64 octets. |
| NAME: SDCertFile  DATA TYPE: String | Conditional | The path of the SD certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: SDKeyFile  DATA TYPE: String | Conditional | The name of the SD certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: SDCAFile  DATA TYPE: String | Conditional | The name of the trusted certificate authority.  The maximum length of the ID string is 256 octets. |

Table 12 describes the sdAssociateResponse JSON object.

Table 12 SD Associate Response Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDName DATA TYPE: string | Required | The name of the sensing device registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: response DATA TYPE: string | Required | The response code for association. |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: heartbeatInterval DATA TYPE: Integer | Required | Heartbeat interval in seconds. |

* + - * 1. SD-TM Capability Information Exchange

Table 13 describes the sdCapabilityRequest JSON object sent by the TM to SD.

Table 13 SD Capability Request Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: sendBaseCapability DATA TYPE: boolean | Conditional | True or False. If false, base capability information is not required. |
| NAME: freqIntervals DATA TYPE: Array of freqInterval | Conditional | Array of freqInterval objects. Each freqInterval object denotes a frequency range as defined in Table 14. |
| NAME: timeIntervals DATA TYPE: Array of timeRange | Conditional | Array of timeInterval objects. Each timeInterval object denotes a time range as defined in Table 15 |
| NAME: scanPeriodicity DATA TYPE: Integer | Conditional | Supported scanPeriodicity interval. The periodicity interval is expressed in number of seconds. |

Table 14 describes the freqInterval JSON object sent by the TM to SD.

Table 14 SCOS Frequency Interval Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: lowFreq DATA TYPE: Integer | Required | The low frequency of a frequency interval. |
| NAME: highFreq  DATA TYPE: Integer | Required | The high frequency of a frequency interval. |

Table 15 describes the timeInterval JSON object sent by the TM to SD.

Table 15 SCOS Time Interval Definition Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: startTime DATA TYPE: Time | Required | The start of a time interval. |
| NAME: endTime  DATA TYPE: Time | Required | The end of a time interval. |

Table 16 SCOS Time Datatype Definition Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: time  DATA TYPE: String | Required | UTC time expressed in the format YYYY-MM-DDThh:mm:ssZ as defined by [1] |

Table 17 describes the sdCapabilityResponse JSON object sent by the SD to TM.

Table 17 SD Capability Response Object Defiintion

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID  DATA TYPE: string | Required | The name of the SD registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SDCapabilityInfo DATA TYPE: sdCapabilityInfo | Conditional | Object describing SD capability (class B SD metadata) as described in Annex B. |
| NAME: freqIntervals DATA TYPE: Array of freqInterval | Conditional | Array of freqInterval objects. Each freqInterval object denotes a frequency range as defined in Table 14. |
| NAME: timeIntervals DATA TYPE: Array of timeRange | Conditional | Array of timeInterval objects. Each timeInterval object denotes a time range as defined in Table 15 |
| NAME: scanPeriodicity DATA TYPE: Integer | Conditional | Supported scanPeriodicity interval. The periodicity interval is expressed in number of seconds. |

* + - * 1. SD-TM Scan Message Exchange

Table 18 describes the sdScanRequest JSON object from TM to SD.

Table 18 SD Scan Request Message Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: TaskID  DATA TYPE: String | Required | Unique ID for the Spectrum Scan.  The maximum length of the ID string is 64 octets. |
| NAME: freqIntervals DATA TYPE: Array of freqInterval | Conditional | Array of freqInterval objects. Each freqInterval object denotes a frequency range as defined in Table 14. |
| NAME: scanResolution DATA TYPE: Integer | Conditional | The suggested frequency resolution for the scan. |
| NAME: TaskDuration  DATA TYPE: number | Required | Duration of scan in milliseconds. |
| NAME: TaskStartTime  DATA TYPE: Time | Required | The start time for the task. |
| NAME: TaskRepeatInterval  DATA TYPE: Number | Optional | The interval in seconds after which the task needs to be repeated. |
| NAME: TaskRepeatCount  DATA TYPE: Number | Optional | The number of times the task needs to be repeated. |
| NAME: TaskEndTime  DATA TYPE: Time | Optional | The end time for the task. |

Table 19 describes the sdScanResponse JSON object from SD to TM.

Table 19 SD Scan Response Message Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: TaskID  DATA TYPE: String | Required | Unique ID for the Spectrum Scan.  The maximum length of the ID string is 64 octets. |
| NAME: scanStatus DATA TYPE: Array of Integer | Required | Array provides scan output status code for each of the freqIntervals. The status code is one of the response codes from Table. The freqIntervals should match with the freqIntervals from the request message. |
| NAME: timestamp DATA TYPE: Time | Required | Timestamp with the associated scanning output. |
| NAME: scanData  DATA TYPE: Array of scanData objects | Required | Array of scanData objects. Each object represents SD measurements for the freqInterval. The scanData is defined in B.3.27.2 |
| NAME: envInfo DATA TYPE: environMetadata | Required | The environmental data including GPS, temperature, and humidity as described in B.3.8 |

Table 20 describes the scanData JSON object sent by the TM to SD.

Table 20 SD Scan Data Message Definition Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: dataFormat DATA TYPE: Integer | Required | The format of the output data as specified in B.3.27.2. |
| NAME: sizeData  DATA TYPE: Integer | Required | The number of measurements. |
| NAME: measData  DATA TYPE: Array of Complex | Required | The complex measurement values. The size of the array is defined by sizeData. |

* + - * 1. SD-TM Heartbeat Message Exchange

Table 21 describes the sdHeartbeatRequest JSON object from TM to SD.

Table 21 SD Heartbeat Request Message Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: calibrate DATA TYPE: boolean | Conditional | If true, SD is required to perform calibration. |
| NAME: calibrateTime DATA TYPE: Time | Conditional | If calibrate true, calibrateTime denotes the time for performing calibration. |

Table 22 describes the sdHeartbeatResponse JSON object from SD to TM.

Table 22 SD Heartbeat Response Message Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: calibrateStatus DATA TYPE: Integer | Conditional | The status code for the scheduled calibration. |
| NAME: envInfo DATA TYPE: envMetadata | Required | The type of DCli.  The maximum length is 64 octets. |
| NAME: healthInfo DATA TYPE: healthMetadata | Required | The SD health metadata as described in B.3.8. |

Table 23describes the healthMetadata JSON object sent by the SD to TM.

Table 23 SD Health Meatadata Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: batteryLevel DATA TYPE: Integer | Required | The battery level in percentage rounded to closest integer. |

* + - * 1. SD-TM Disassociation Message Exchange

Table 24describes the sdDisassociateRequest JSON object from SD to TM.

Table 24 SD Disassociate Request Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID  DATA TYPE: string | Required | The ID assigned to SD by the SCOS operator.  The maximum length is 64 octets. |
| NAME: SDName DATA TYPE: string | Required | The name of the sensing device registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |

Table 25 describes the sdDisassociateResponse JSON object from TM to SD.

Table 25 SD Disassociate Response Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDName  DATA TYPE: string | Required | The name of the SD registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: string | Required | The response code for dissociation request. |
| NAME: oldSDID  DATA TYPE: string | Required | The SD ID that has been dissociated  The maximum length is 64 octets. |

DCli<>TM Messages

Following table describes the SCOS Data Request API methods

Table 26 Messages Exchanged Between DCli and TM

|  |  |  |
| --- | --- | --- |
| scos\_method\_name | JSON Array Name of Request Message | JSON Array Name of Response Message |
| “ta\_associate” | *taAssociateRequest* | *taAssociateResponse* |
| “ta\_resource\_discovery” | *taResourceDiscoveryRequest* | *taResourceDiscoveryResponse* |
| “ta\_schedule\_scan” | *taScheduleScanRequest* | *taScheduleScanResponse* |
| “ta\_scan\_status” | *taScanStatusRequest* | *taScanStatusResponse* |
| “ta\_scan\_notify” | *taScanNotification* | *taScanNotificationResponse* |
| “ta\_dissociate” | *taDissociateRequest* | *taDissociateResponse* |

* + - * 1. DCli-TM Association Message Exchange

Table 27 describes the taAssociateRequest JSON object from DCli to TM.

Table 27 DCli Associate Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliName  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: SMName  DATA TYPE: string | Required | The name of the sensing manager to associate with.  The maximum length is 64 octets. |
| NAME: DCliType DATA TYPE: string | Required | The type of DCli. Valid values include {“DCliTypeA”, “DCliTypeB”, “DCliTypeC”} |
| NAME: DCliID DATA TYPE: string | Optional | The unique ID assigned to the Data Client. The maximum length is 64 octets. |
| NAME: DCliCertFile  DATA TYPE: String | Optional | The path of the DCli certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DCliKeyFile  DATA TYPE: String | Optional | The name of the DCli certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DCliCAFile  DATA TYPE: String | Optional | The name of the trusted certificate authority.  The maximum length of the ID string is 256 octets. |
| NAME: admPreferences  DATA TYPE: AdmPreferences | Optional | A DCli can optionally specify certain preferences related to how scanning task administration is performed. |

Table 28 describes the AdmPreferences object

Table 28 DCli Administrative Preferences

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: useNewTaskforRestart  DATA TYPE: boolean | Optional | DCli chooses to have new TaskID for a restarted task. In the case of restarting a sensing task, TM provides old TaskID as well in order to associate the two. More details in the Section 7 administration. |
| NAME: useNewTaskforMigration  DATA TYPE: boolean | Optional | DCli chooses to have new TaskID for a sensing task with change in SD resource. In the case of migrating a sensing task, TM provides old TaskID as well in order to associate the two. More details in the Section 7 administration. |

Table 29 describes the taAssociateResponse JSON object from TM to DCli

Table 29 DCli Associate Response Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliName  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SMName  DATA TYPE: string | Required | The name of the sensing manager to associate the DCli with.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: string | Required | The response code for the DCli association request.  The maximum length is 64 octets. |
| NAME: DCliID DATA TYPE: string | Required | The unique ID assigned to the Data Client. The maximum length is 64 octets. |
| NAME: SMID DADCli TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |

* + - * 1. DCli-TM Resource Discovery Message Exchange

Table 30 describes the DCliResourceDiscoveryRequest JSON object from DCli to TM.

Table 30 DCli Resource Discovery Request

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliID  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: perSDInfoRequest  DATA TYPE: Boolean | Required | If True, per SD resource/capability information is requested. |
| NAME: freqIntervals DATA TYPE: Array of freqInterval | Optional | Array of freqInterval objects. Each freqInterval object denotes a frequency range as defined in Table 14. |
| NAME: scanDataFormat DATA TYPE: Array of timeInterval | Optional | The format of the scan data as described in B.3.27.2.  The maximum length is 64 octets. |
| NAME: scanResolution DATA TYPE: Integer | Optional | The minimum desired scan resolution. |
| NAME: locations  DATA TYPE: Array of Location objects | Optional | Array of Location objects for the specified scan frequencies. Each Location object denotes desired coordinates as defined in B.3.9. |
| NAME: locationAccuracy  DATA TYPE: Integer | Optional | Desired accuracy for location in terms of maximum distance in meters from the specified coordinate. |

Table 31 describes the location for the DCliResourceDiscoveryResponse JSON object from TM to DCli.

Table 31 DCli Resource Discovery Location Descriptor Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: Latitude  DATA TYPE: string | Required | Latitude is expressed in format DD°MM’SS’’ N/S  The maximum length is 64 octets. |
| NAME: Longitude DATA TYPE: string | Required | Longitude is expressed in format DD°MM’SS’’ W/E  The maximum length is 64 octets. |

Table 32 describes the DCliResourceDiscoveryResponse JSON object from TM to DCli.

Table 32 DCli Resource Discovery Response Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SMID DATA TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |
| NAME: DCliID  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: sdResourceInfo  DATA TYPE: Array of sdCapabilityInfo objects | Conditional | If perSDInfoRequest is true in the request, array of sdCapabilityInfo (Annex B) objects is included. |
| NAME: statusFreqIntervals DATA TYPE: Array of Integer | Conditional | Status codes for each of the freqIntervals from the request message that meet the scanDataformat and scanResolution. |
| NAME: AccuracyLocation  DATA TYPE: Array of Integer | Conditional | Accuracy for each of the Locations from the request message in terms of distance (measured in meter). |

* + - * 1. DCli-TM Scan Request Message Exchange

Table 33 describes the DCliScheduleScanRequest JSON object from DCli to TM.

Table 33 DCli Schedule Scan Request Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliID  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: freqIntervals DATA TYPE: Array of freqInterval | Required | Array of freqInterval objects. Each freqInterval object denotes a frequency range as defined in Table 14. |
| NAME: scanDataFormat DATA TYPE: Array of timeInterval | Required | The format of the scan data as described in B.3.27.2.  The maximum length is 64 octets. |
| NAME: scanResolution DATA TYPE: Integer | Required | The minimum desired scan resolution. |
| NAME: locations  DATA TYPE: Array of Location objects | Required | Array of Location objects. Each Location object denotes desired coordinates as defined in B.3.9. |
| NAME: locationAccuracy  DATA TYPE: Integer | Required | Desired accuracy for location in terms of maximum distance in meters from the specified coordinate. |

Table 34 describes the DCliScheduleScanResponse JSON object from TM to DCli

Table 34 DCli Schedule Scan Response Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SMID DATA TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |
| NAME: DCliID  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: DCliScanID  DATA TYPE: string | Required | The unique ID assigned for the scan scheduled for the DCli.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: Array of Integer | Required | Status codes for each of the scan frequency ranges that support desired scan parameters and the desired locationAccuracy. |

* + - * 1. DCli-TM Scan Status Inquiry Message Exchange

Table 35 describes the sdScanStatusRequest JSON object from DCli to TM.

Table 35 DCli Scan Status Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliID  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SMID DATA TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |
| NAME: DCliScanID  DATA TYPE: string | Required | The unique ID assigned for the scan scheduled for the DCli.  The maximum length is 64 octets. |

Table 36 describes the sdScanStatusResponse JSON object from TM to DCli

Table 36 DCli Scan Status Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliID  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SMID DATA TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |
| NAME: DCliScanID  DATA TYPE: string | Required | The unique ID assigned for the scan scheduled for the DCli.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: Array of Integer | Required | Status codes for each of the scan frequency ranges that support desired scan parameters and the desired locationAccuracy. |

* + - * 1. DCli-TM Scan Notification Message Exchange

Upon completion of the scan or upon error event, TM notifies the DCli of the status for the scan.

Table 37 describes the sdScanNotifyRequest JSON object from DCli to TM.

Table 37 SD Scan Notification Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliID  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SMID DATA TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |
| NAME: DCliScanID  DATA TYPE: string | Required | The unique ID assigned for the scan scheduled for the DCli.  The maximum length is 64 octets. |
| NAME: status  DADCli TYPE: Array of Integer | Required | Status codes for each of the scan frequency ranges that support desired scan parameters and the desired locationAccuracy. |

* + - * 1. DCli-TM Dissociation Message Exchange

Table 38 describes the DCliDissociateRequest JSON object from DCli to TM.

Table 38 DCli Dissociation Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliID  DATA TYPE: string | Required | The ID assigned to DCli by the SCOS operator.  The maximum length is 64 octets. |
| NAME: DCliName DATA TYPE: string | Required | The name of the DCli registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |

Table 39 describes the sdDissociateResponse JSON object from TM to DCli

Table 39 DCli Dissociation Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCliName  DATA TYPE: string | Required | The name of the Data Client registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: string | Required | The response code for dissociation request. |
| NAME: oldDCliID  DATA TYPE: string | Required | The DCli ID that has been dissociated  The maximum length is 64 octets. |

SD<>DM Messages

Table 40 denotes the messages exchanged between SD and DM.

Table 40 Messages Exchanged Between SD and DM

|  |  |  |
| --- | --- | --- |
| scos\_method\_name | JSON Array Name of Request Message | JSON Array Name of Response Message |
| “sd\_dm\_associate” | *sdAssociateRequest* | *sdAssociateResponse* |
| “sd\_dm\_publish” | *sdPublishRequest* | *sdPublishResponse* |
| “sd\_dm\_heartbeat” | *sdHeartbeatRequest* | *sdHeartbeatResponse* |
| “sd\_dm\_disassociate” | *sdDisassociateRequest* | *sdDisassociateResponse* |

* + - * 1. SD-DM Association Message Exchange

Table 41 describes the sdAssociateRequest JSON object from SD to DM.

Table 41 SD Associate Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDName DATA TYPE: string | Required | The name of the sensing device registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: response DATA TYPE: string | Required | The response code for association. |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: heartbeatInterval DATA TYPE: Integer | Required | Heartbeat interval in seconds. |

* + - * 1. SD-DM Publish Message

Table 42 describes the sdPublishRequest JSON object from SD to DM.

Table 42 SD Publish Sensing Data Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: TaskID  DATA TYPE: String | Required | Unique ID for the Spectrum Scan.  The maximum length of the ID string is 64 octets. |
| NAME: scanStatus DATA TYPE: Array of Integer | Required | Array provides scan output status code for each of the freqIntervals. The status code is one of the response codes from Table. The freqIntervals should match with the freqIntervals from the request message. |
| NAME: timestamp DATA TYPE: Time | Required | Timestamp with the associated scanning output. |
| NAME: scanData  DATA TYPE: Array of scanData objects | Required | Array of scanData objects. Each object represents SD measurements for the freqInterval. The scanData is defined in B.3.27.2 |
| NAME: envInfo DATA TYPE: environMetadata | Required | The environmental data including GPS, temperature, and humidity as described in B.3.8 |

Table 43 describes the scanData JSON objects from SD to DM.

Table 43 SD ScanData Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: dataFormat DATA TYPE: Integer | Required | The format of the output data as specified in B.3.27.2. |
| NAME: sizeData  DATA TYPE: Integer | Required | The number of measurements. |
| NAME: measData  DATA TYPE: Array of Complex | Required | The complex measurement values. The size of the array is defined by sizeData. |

Table 44 describes the sdPublishResponse JSON object from SD to DM.

Table 44 SD Publish Sensing Data Response Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: TaskID  DATA TYPE: String | Required | Unique ID for the Spectrum Scan.  The maximum length of the ID string is 64 octets. |
| NAME: status DATA TYPE: Array of Integer | Required | Each entry shows status for the publish request of the scanning data for each of the freqIntervals. |
| NAME: timestamp DATA TYPE: Time | Required | Timestamp for the associated scanning data that DM is acknowledging. |

* + - * 1. SD-DM Heartbeat Message

Table 45Table 45 describes the heartbeat message exchanged between SD and DM

Table 45 SD Heartbeat Request Message With DM

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: Cmd  DATA TYPE: Time | Optional | If nonzero, SD is given a command. Currently, two command codes are defined. If cmd is 1, DM is asking SD to list all topics for which the SD is publishing. If cmd is 2, DM is asking to stop publishing for a topic or all topics as suggested by next object in the message. |
| NAME: topic  DATA TYPE: string | Conditional | If cmd is 2, this field denotes the topic for which DM is asking SD to stop publishing. If no topic is specified, DM is asking to stop publishing for all the topics. |

Table 46 denotes the topic details exchanged in the heartbeat response message from SD and DM.

Table 46 Topic Details Exchange in Heartbeat Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: activeTopics DATA TYPE: Array of string | Required | Each entry in the list describes an active topic. The maximum length is 64 octets. |

* + - * 1. SD-DM Dissociation Message Exchange

Table 48 describes the sdDissociateRequest JSON object from SD to DM.

Table 47 SD Dissociate Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDID  DATA TYPE: string | Required | The ID assigned to SD by the SCOS operator.  The maximum length is 64 octets. |
| NAME: SDName DATA TYPE: string | Required | The name of the sensing device registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |

Table 48 describes the sdDisassociateResponse JSON object from DM to SD.

Table 48 SD Dissociate Response Object

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDName  DATA TYPE: string | Required | The name of the SD registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: string | Required | The response code for dissociation request. |
| NAME: oldSDID  DATA TYPE: string | Required | The SD ID that has been dissociated  The maximum length is 64 octets. |

DCon<>DM Messages

Following Table 49 enumerates the messages exchanged between DM and DCon.

Table 49Messages Exchanged Between DCon and DM

|  |  |  |
| --- | --- | --- |
| scos\_method\_name | JSON Array Name of Request Message | JSON Array Name of Response Message |
| “dc\_associate” | *dcAssociateRequest* | *dcAssociateResponse* |
| “dc\_subscribe” | *dcSubscribeRequest* | *dcSubscribeResponse* |
| “dc\_topicdata” | *dcTopicData* | *dcTopicDataResponse* |
| “dc\_unsubscribe” | *dcUnSubscribeRequest* | *dcUnSubscribeResponse* |
| “dc\_heartbeat” | *dcHeartbeatRequest* | *dcHeartbeatResponse* |
| “dc\_disassociate” | *dcDisassociateRequest* | *dcDisassociateResponse* |

* + - * 1. DCon-DM Association Message Exchange

Table 50 describes the dcAssociateRequest JSON object from DC to DM.

Table 50 DCon Associate Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCName  DATA TYPE: string | Required | The name of the data-consumer registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator  DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: DMName  DATA TYPE: string | Required | The name of the data manager to associate with.  The maximum length is 64 octets. |
| NAME: DCID  DATA TYPE: string | Optional | The unique ID assigned to the data-client. The maximum length is 64 octets. |
| NAME: DCCertFile  DATA TYPE: String | Optional | The path of the DCli certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DCKeyFile  DATA TYPE: String | Optional | The name of the DC certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: DCCAFile  DATA TYPE: String | Optional | The name of the trusted certificate authority.  The maximum length of the ID string is 256 octets. |

Table 51 describes the dcAssociateResponse JSON object from DM to DCon

Table 51 DCon Associate Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCName  DATA TYPE: string | Required | The name of the DCli registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: DMName  DATA TYPE: string | Required | The name of the data manager to associate the DC with.  The maximum length is 64 octets. |
| NAME: SCOSOperator  DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: string | Required | The response code for the DCli association request. The maximum length is 64 octets. |
| NAME: DCID  DATA TYPE: string | Required | The unique ID assigned to the Data Client. The maximum length is 64 octets. |
| NAME: DMID  DATA TYPE: string | Required | The unique ID of the data manager. The maximum length is 64 octets. |

* + - * 1. DCon-DM Subscribe Message Exchange

Table 52 describes the dcSubscribeRequest JSON object from DC to DM.

Table 52 DCon Subscribe Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCID  DATA TYPE: string | Required | The unique ID assigned to the DCon. The maximum length is 64 octets. |
| NAME: TopicID  DATA TYPE: String | Required | Unique TopicID for the spectrum sensing data to associate the DC with.  The maximum length of the ID string is 128 octets. |
| NAME: TAID  DATA TYPE: String | Required | ID of Data Client associated with the scan.  The maximum length of the ID string is 64 octets. |

Table 53 describes the dcSubscribeResponse JSON object from DM to DC.

Table 53 DCon Subscribe Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCID  DATA TYPE: string | Required | The unique ID assigned to the DCon. The maximum length is 64 octets. |
| NAME: TopicID  DATA TYPE: String | Required | Unique TopicID for the spectrum sensing data to associate the DC with.  The maximum length of the ID string is 128 octets. |
| NAME: status  DATA TYPE: Integer | Required | Response code to subscribe request. |

* + - * 1. DCon-DM TopicData Message Exchange

Table 54 describes the dcTopicData JSON object from DM to DM.

Table 54 DCon Topic Data Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCID  DATA TYPE: String | Required | The unique ID assigned to the DCon. The maximum length is 64 octets. |
| NAME: TaskID  DATA TYPE: String | Required | Unique ID for the Spectrum Scan.  The maximum length of the ID string is 64 octets. |
| NAME: scanStatus  DATA TYPE: Array of Integer | Required | Array provides scan output status code for each of the freqIntervals. The status code is one of the response codes from Table. The freqIntervals should match with the freqIntervals from the request message. |
| NAME: timestamp DATA TYPE: Time | Required | Timestamp with the associated scanning output. |
| NAME: scanData  DATA TYPE: Array of scanData objects | Required | Array of scanData objects. Each object represents SD measurements for the freqInterval. The scanData is defined in B.3.27.2 |
| NAME: envInfo  DATA TYPE: environMetadata | Required | The environmental data including GPS, temperature, and humidity as described in B.3.8 |

Table 55 describes the scanData JSON object from DM to DCon.

Table 55 DM Published Data Object Definition

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: dataFormat  DATA TYPE: Integer | Required | The format of the output data as specified in B.3.27.2. |
| NAME: sizeData  DATA TYPE: Integer | Required | The number of measurements. |
| NAME: measData  DATA TYPE: Array of Complex | Required | The complex measurement values. The size of the array is defined by sizeData. |

Table 56 describes the dcTopicDataResponse JSON object from DCon to DM.

Table 56 Topic Data Acknowledgement Message from DCon

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCID  DATA TYPE: string | Required | The unique ID assigned to the Data Consumer.  The maximum length is 64 octets. |
| NAME: TopicID  DATA TYPE: String | Required | Unique ID for the Topic.  The maximum length of the ID string is 128 octets. |
| NAME: status  DATA TYPE: Array of Integer | Required | Each entry shows status for the TopicData request of the scanning data for each of the freqIntervals. |
| NAME: timestamp DATA TYPE: Time | Required | Timestamp for the associated scanning data that DC is acknowledging. |

* + - * 1. DC-DM Unsubscribe Message Exchange

Table 57 describes the dcUnSubscribeRequest JSON object from DC to DM.

Table 57 DCon Unsubscribe Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCID  DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: TopicID  DATA TYPE: String | Required | Unique TopicID for the spectrum sensing data the DC wants to unsubscribe.  The maximum length of the ID string is 128 octets. |
| NAME: TAID  DATA TYPE: String | Required | ID of Data Client associated with the scan.  The maximum length of the ID string is 64 octets. |

Table 58 describes the dcUnSubscribeResponse JSON object from DM to DC.

Table 58 Dcon Unsubscribe Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCID  DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: TopicID  DATA TYPE: String | Required | Unique TopicID for the spectrum sensing data the DC wants to unsubscribe.  The maximum length of the ID string is 128 octets. |
| NAME: status  DATA TYPE: Integer | Required | Response code to unsubscribe request. |

* + - * 1. DC-DM Heartbeat Message

Table 59describes the dcHeartbeatRequest JSON object from DM to DC.

Table 59 DCon Heartbeat Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCID  DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: Info  DATA TYPE: Integer | Optional | If DM intends to notify DC certain information related to specific topic or DM/connectivity specific health information.  Information codes:  0-15: DM/connectivity specific information  >15: Topic specific information |
| NAME: topic  DATA TYPE: string | Conditional | If information code > 15, this field denotes the topic for which DM is providing additional information. |

Table 60 describes the dcHeartbeatResponse JSON object from DC to DM.

Table 60 DCon Heartbeat Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCID DATA TYPE: string | Required | The unique ID assigned to the data-client. The maximum length is 64 octets. |
| NAME: topicsNeedAttention DATA TYPE: Array of string | Optional | Each entry in the list describes an active topic that needs attention.  The maximum length of each topic entry is 128 octets. |

* + - * 1. DC-DM Dissociation Message Exchange

Table 61 describes the dcDissociateRequest JSON object from DC to DM.

Table 61 DCon Dissociation Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DCID  DATA TYPE: string | Required | The ID assigned to DC by the SCOS operator.  The maximum length is 64 octets. |
| NAME: DCName  DATA TYPE: string | Required | The name of the DC registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator  DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |

Table 62 describes the dcDissociateResponse JSON object from DM to DC

Table 62 DCon Dissociation Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: TAName  DATA TYPE: string | Required | The name of the DC registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: string | Required | The response code for dissociation request. |
| NAME: oldDCID  DATA TYPE: string | Required | The DC ID that has been dissociated  The maximum length is 64 octets. |

TM<>DM Messages

The SCOS control plane is managed by the TM and the data plane is managed by DM. The DM and TM communicate in order to synchronize the control plane and data plane. Table 63 describes the messages exchanged between the TM and the DM.

Table 63 Messages Exchanged Between TM and DM

|  |  |  |
| --- | --- | --- |
| scos\_method\_name | JSON Array Name of Request Message | JSON Array Name of Response Message |
| “sm\_dm\_associate” | *smAssociateRequest* | *smAssociateResponse* |
| “sm\_task\_coordinate” | *smTaskCoordinationRequest* | *smTaskCoordinationResponse* |
| “sm\_task\_moderation” | *smTaskModerationRequest* | *smTaskModerationResponse* |
| “sm\_dm\_heartbeat” | *smHeartbeatRequest* | *smHeartbeatResponse* |
| “sm\_dm\_disssociate” | *smDisassociateRequest* | *smDisassociateResponse* |

* + - * 1. TM-DM Association Message Exchange

Table 64 describes the smAssociateRequest JSON object from DC to DM.

Table 64 DM Association Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SMName  DATA TYPE: string | Required | The name of the sensing manager registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator  DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: DMName  DATA TYPE: string | Required | The name of the data manager to associate with.  The maximum length is 64 octets. |
| NAME: SMID  DATA TYPE: string | Optional | The unique ID assigned to the sensing manager.  The maximum length is 64 octets. |
| NAME: SMCertFile  DATA TYPE: String | Optional | The path of the TM certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: SMKeyFile  DATA TYPE: String | Optional | The name of the TM certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: SMCAFile  DATA TYPE: String | Optional | The name of the trusted certificate authority.  The maximum length of the ID string is 256 octets. |

Table 65 describes the smAssociateResponse JSON object from TM to DM

Table 65 DM Associate Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDName  DATA TYPE: string | Required | The name of the sensing manager registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: DMName  DATA TYPE: string | Required | The name of the data manager to associate the SD with.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: string | Required | The response code for the TM-DM association request.  The maximum length is 64 octets. |
| NAME: DMID DATA TYPE: string | Required | The unique ID of the data manager. The maximum length is 64 octets. |
| NAME: SMID DATA TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |

* + - * 1. TM-DM Sensing Task Coordination Message Exchange

Table 66 describes the dmTaskCoordinationRequest JSON object from TM to DM.

Table 66 DM Task Coordination Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SMID DATA TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |
| NAME: TaskID  DATA TYPE: String | Required | Unique TaskID for the spectrum sensing data to associate the DM with.  The maximum length of the ID string is 128 octets. |
| NAME: DMID DATA TYPE: String | Required | ID of data manager associated with the scan.  The maximum length of the ID string is 64 octets. |

Table 67 describes the dmTaskCoordinationResponse JSON object from DM to TM.

Table 67 DM Task Coordination Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DMID DATA TYPE: string | Required | The unique ID assigned of the DM handling the sensing data distribution for the sensing task. The maximum length is 64 octets. |
| NAME: TaskID  DATA TYPE: String | Required | Unique TaskID for the spectrum sensing task.  The maximum length of the ID string is 128 octets. |
| NAME: status DATA TYPE: Integer | Required | Response code to subscribe request. |

* + - * 1. TM-DM TopicData Message Exchange

Table 68 describes the dmTaskModeration JSON object from DM to TM.

Table 68 DM Task Moderation Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SMID  DATA TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |
| NAME: DMID  DATA TYPE: String | Required | ID of data manager associated with the scan.  The maximum length of the ID string is 64 octets. |
| NAME: TaskID  DATA TYPE: String | Required | Unique TaskID for the spectrum sensing data to associate the DM with.  The maximum length of the ID string is 128 octets. |
| NAME: ModCmd DATA TYPE: Integer | Required | Sensing task moderation command code. Following are the defined command codes.  0: Invalid cmd  1: Data validation errors  2: Data rate mismatch  3: Switch the sensing task to another DM  4-31: reserved commands  >31: custom command codes |

Table 69 describes the dmTaskModerationResponse JSON object from TM to DM.

Table 69 DM Task Moderation Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DMID DATA TYPE: string | Required | The unique ID assigned of the DM handling the sensing data distribution for the sensing task. The maximum length is 64 octets. |
| NAME: TaskID  DATA TYPE: String | Required | Unique TaskID for the spectrum sensing task.  The maximum length of the ID string is 128 octets. |
| NAME: status DATA TYPE: Integer | Required | Response code for the task moderation request. |

* + - * 1. TM-DM Heartbeat message

Table 70 describes the dmHeartbeatRequest JSON object from DM to TM.

Table 70 DM Heartbeat Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SMID  DATA TYPE: string | Required | The unique ID of the sensing manager. The maximum length is 64 octets. |
| NAME: DMID  DATA TYPE: string | Required | The unique ID of the data manager. The maximum length is 64 octets. |
| NAME: Info  DATA TYPE: Integer | Optional | If DM intends to notify DC certain information related to specific topic or TM/connectivity specific health information.  Information codes:  0-15: TM/connectivity specific information  >15: Topic specific information |
| NAME: topic  DATA TYPE: string | Conditional | If information code > 15, this field denotes the topic for which TM is providing additional information. |

Table 71 describes the dmHeartbeatResponse JSON object from TM to DM.

Table 71 DM - TM Heartbeat Exchange

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: DMID  DATA TYPE: string | Required | The unique ID assigned to the data-client. The maximum length is 64 octets. |
| NAME: topicsNeedAttention DATA TYPE: Array of string | Optional | Each entry in the list describes an active topic that needs attention.  The maximum length of each topic entry is 128 octets. |

* + - * 1. TM-DM Dissociation Message Exchange

Table 72 describes the dmDissociateRequest JSON object from DC to DM.

Table 72 DM Dissociate Request Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SMID  DATA TYPE: string | Required | The ID of the sensing manager.  The maximum length is 64 octets. |
| NAME: SMName  DATA TYPE: string | Required | The name of the sensing manager.  The maximum length is 64 octets. |
| NAME: DMID  DATA TYPE: string | Required | The ID of the data manager.  The maximum length is 64 octets. |
| NAME: DMName  DATA TYPE: string | Required | The name of the data manager.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |

Table 73 describes the dmDissociateResponse JSON object from DM to DC

Table 73 DM Dissociate Response Message

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SMName  DATA TYPE: string | Required | The name of the DC registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: DMName  DATA TYPE: string | Required | The name of the data manager.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: status  DATA TYPE: string | Required | The response code for dissociation request. |

1. System Administration and Security
   1. Administration

Administrative functions on the SDs and SSMs are largely assumed to be implementer-specific, and out of scope for the standard, but recommendations are included in the informative annex.

This interface shall have a secure mechanism to administer the system, allowing:

* performing of SD calibrations
* updating firmware of SDs
* changing configuration of SD, and making associated changes to the SD configuration file
* triggering SD reboots and other SD hardware maintenance functions.

Additionally, administrator shall have a mechanism allowing

* pause or delete particular scanning task and corresponding message-queue/topic.
* flush a message-queue and restart task with the existing/new taskID
* dissociate a DCli and all scheduled/ongoing tasks from the DCli
* dissociate a SD
* migrate all sensing schedules for an SD to other SD(s).

The administration interface must be a secure interface with key exchange, and these keys must not be the same as keys used for Data Client<>TM<>SD authentication.

The standard suggests using special administration mode (AdminCmd and AdminCmdResponse) in the message header. There may be multiple response messages for a single command indicating the execution status of the command such as scheduled, in-progress, and done. Following table identifies administration commands for various interfaces.

SD Administration

Table 74 provides Summary of the TM-SD Interface.

Table 74 TM -SD Administrative Interface Summary

|  |  |  |  |
| --- | --- | --- | --- |
| Administration Command | SD Action | Administration Response | Details |
| NAME: admCalibrate  DATA TYPE: String | SD performs CALif no time specified else schedules CAL. | NAME: admCalibrateStatus  DATA TYPE: String | The time may be optionally specified with this command. The status from the SD could be SD\_CAL\_IN\_PROGRESS, SD\_CAL\_SCHEDULED, SD\_CAL\_DONE, or SD\_CAL\_ERR |
| NAME: admFwUpdate  DATA TYPE: String | SD updates firmware | NAME: admFwUpdateStatus  DATA TYPE: String | The TM provides the firmware path, name, and update time. The SD status could be SD\_FW\_UPDATE\_SCHEDULED, SD\_FW\_UPDATE\_IN\_PROGRESS, SD\_FW\_UPDATE\_DONE, or SD\_FW\_UPDATE\_ERR |
| NAME: admConfig  DATA TYPE: String | SD applies suggested configuration | NAME: admConfigStatus  DATA TYPE: String | The TM provides the config path, filename, and update time. The SD status could be SD\_CONF\_UPDATE\_SCHEDULED, SD\_CONF\_UPDATE\_IN\_PROGRESS, SD\_CONF\_UPDATE\_DONE, or SD\_CONF\_UPDATE\_ERR |
| NAME: admPowerCycle  DATA TYPE: String | SD performs power cycle | NAME: admPowerCycleStatus  DATA TYPE: String | The TM optionally provides time. The SD status could be SD\_POWER\_CYCLE\_SCHEDULED, SD\_POWER\_CYCLE\_DONE, or SD\_POWER\_CYCLE\_ERR |

In addition to these commands, it is possible to define custom commands for administration. The standard may include more common use-cases in its next revision.

Sensing Task Administration

* + - * 1. Pause/Resume a sensing task

It may be essential to pause a sensing task due to administrative or technical issue. In this case, a message-exchange is initiated from the TM to notify corresponding DCli and SDs. TM coordinates this action with DM and a message exchange is initiated from DM to DCs. The SCOS administrator may need to resume the sensing task after the issue is resolved.

* + - * 1. Restart a sensing task

It may be essential to restart a sensing task due to administrative or technical issue. Here, in certain use-cases, TAs may prefer that a scanTaskID generated. TAs may communicate this preference during association (useNewTaskforRestart). If DCli prefers a new task ID, old task ID is also communicated for associating the two scanning tasks.

A message-exchange is initiated from the TM to notify corresponding DCli and SDs. TM coordinates this action with DM and a message exchange is initiated from DM to DCs.

* + - * 1. Migrate all the sensing tasks associated with an SD

It may be essential to migrate all sensing tasks to other SCOS resources (due to technical or security issue) . TM may assign a new task ID depending on TAs preferences.

A message-exchange is initiated from the TM to notify corresponding DCli and SDs. TM coordinates this action with DM and a message exchange is initiated from DM to DCs.

* + - * 1. Delete a sensing task

It may be essential to delete a sensing task (due to technical, resource, or security issue).

A message-exchange is initiated from the TM to notify corresponding DCli and SDs. TM coordinates this action with DM and a message exchange is initiated from DM to DCs.

* + - * 1. Delete all sensing tasks from a DCli

It may be essential to delete a sensing task (due to technical, resource, or security issue) .

A message-exchange is initiated from the TM to notify corresponding DCli and SDs. TM coordinates this action with DM and a message exchange is initiated from DM to DCs.

SCOS Platform Behavior Specification

In addition to explicit administration, SCOS operator could define behavior of SCOS platform using policy construct. Following are a few examples of specifying SCOS platform behavior

* Allowed data distribution modes - streaming and/or store-forward
* Protocol choice for each of the interfaces - An SCOS operator may prefer MQTT for DM-DC interface and another operator may choose RESTful HTTP interface.
* Security mode for each of the interfaces - An SCOS operator may choose to enforce secure transport for all the interfaces (SD-DM, SD-TM, DM-TM, DCli-TM, and DM-DC)
  1. Security Systems

Scope

The SCOS standard includes security measures toward the maintaining the integrity and confidentiality of the sensing tasks and sensing data. Also, SCOS standard includes measures for ensuring authenticity of the messages. The standard makes provision for the security features and these are highly recommended however, it is upto the SCOS administrator to enforce the security mechanisms on most interfaces.For TM-DM interface, the SCOS platform must include security mechanisms for maintaining the integrity and confidentiality of the sensing tasks and sensing data.

Another part of platform security is authorization. Administrators need to ensure that only authorized users can issue sensing tasks, only authorized users have access to the sensing data published by the platform, and only authorized users can issue certain privileged scans. The standard in the current draft provides an approach to implementing administrative policies. Currently, in this version of the standard, The approach is not a mandatory approach and SCOS administrators may choose to implement policy using an alternate approach.

* + - * 1. Out of scope

Following security aspects are out of scope of the standard

* Physical security of the infrastructure (SDs, TM, DM)
* Availability of the sensing data - The sensing tasks would typically generate enormous amount of sensing data and the SCOS standard does not require the SCOS platform implementation to make sensing data available past it has been received by the data consumers. If data consumers happen to lose the data, sensing needs to be performed again.
* Redundancy model for TM and DM - TM and DM are key entities and administrators may include a redundancy model for TM and DM to improve SCOS platform availability. The redundancy model is out of scope of the standard,

Authentication, Confidentiality, and Integrity

The standard requires implementing following procedure on TM-DM interface. On the remaining interfaces, it is highly recommended.

* TLS mutual authentication shall be performed per [n.1]
* EnityA (For example,DCli) communicates with EntityB (For example, TM). TLS-v1.2 as specified in [n.3] shall be used to perform authentication. Previous versions of TLS (e.g., TLS-v1.1 per RFC-4346, TLS-v1.0 per RFC-2246 or SSL-v3.0) shall not be used.
* During the TLS exchange, mutual authentication shall be performed. The EntityA (For example, DCli) initiating the TLS connection shall authenticate the EntityB (For example, TM), and the EntityB (For example, TM) shall authenticate the EntityA (For example, DCli).
* During the TLS message exchange, the EntityA (For example, DCli) shall authenticate EntityB (For example, TM) according to the procedures defined in [n.4]. Server certificate validation shall be performed according to the procedures in [n.5].
* A EntityA (For example, DCli) which is unable to successfully authenticate an EntityB (For example, TM) shall abort the TLS connection establishment procedure. It is implementation specific when the EntityA (For example, DCli) should re-attempt the TLS connection establishment procedure.
* During the TLS message exchange, the EntityA (For example, DCli) provides its client certificate to the EntityB (For example, TM). The EntityB (For example, TM) shall perform client certificate validation according to the procedures in [n.5]. The EntityB (For example, TM) which is unable to successfully authenticate a EntityA (For example, DCli) shall abort the TLS connection establishment procedure.

Authorization

The standard suggests implementing authorization using policy construct for SCOS control plane and data plane.

* + - * 1. Control Plane Authorization

Control plane authorization is implemented at the TM. It includes the

* ability to enforce a regulatory policy to determine if the location, time, frequency specified in the requested scan are compliant with regulations
* ability to check if the user is authorized to issue the requested sensing task
* ability to define priority for all scans from a specific user
* ability to define priority for a certain set of SD resources
  + - * 1. Data Plane Authorization

Data plane authorization is implemented at the DM. It includes the

* ability to check if the data consumer is authorized to subscribe to the requested sensing data.
* rules for how data is distributed for various data consumers. Only privileged data consumers may be able to request store and forward interface.
* ability to define max storage space in case of store and forward mode

The SCOS security mechanisms and certain administration mechanisms (as described in Section 7.1.3) are implemented using a policy file which is securely installed by the SCOS operator.

# Informative: Reference Applications

## White Space device radio operation

Either the network operator or device operator using spectrum sensing to identify primary or other secondary users of particular channels. Spectrum sensing would either built into the radio devices or in standalone sensing units.

The standard allows a “CR Mode” of operation that would make it suitable for use in radio systems to complement Geolocation Databases (such as a WSDB).

## National spectrum regulation

National radio regulators would use a system comprising spectrum sensing devices to feed into a national spectrum utilization database for assignment management and planning purposes, and generating historical records for compliance monitoring and enforcement.

Devices deployed in various scenarios:

* Fixed devices at key locations and high sites
* Mobile devices on vehicles that travel widely and can create a sample set of spectrum utilization through snapshots at time or location intervals
* Devices either at fixed locations or periodically moved to create location-based spectrum utilization datasets
* Nationally deployed in a swarm of a given device density to create real-time national spectrum utilization maps and for validation of Spectrum Geolocation databases.

When spectrum monitoring is used for automated spectrum usage enforcement, data from a spectrum monitoring system has a critical role in the six basic steps for spectrum enforcement:

1. detecting, 2. identifying and classifying, 3. locating, 4. reporting, 5. mitigating, 6. remediating interference.

It is important to note that each of these six steps may, in general, require a different data type to be collected and stored; ranging from amplitude only information to raw IQ samples. It is possible for the sensing network to process spectrum data at the edge and only report the result of the processing, where conceivably a sensor or set of sensors can identify and locate an emitter without sending the raw spectrum measurements.

For example, the sensing network might report and store only the location information along with the signal classification information. This standard has been made flexible enough to define and enable both the collection of the various data types, along with associated meta-data, as well as the reporting and collection of the results of data analysis performed at the edge.

Spectrum management systems work to accomplish agency missions in geographic area(s) with limited and often shrinking frequency assignments. Monitoring can support spectrum managers in being more efficient by providing real-time and historical information about the RF environment on-base and at-boundary. Further, monitoring information can be used to mitigate and protect government wireless assets from intentional and unintentional interference.

## Research programs

Scientists using sensitive radio frequency systems (e.g. radio-telescopes) struggle with RF interference. SCOS devices can let them identify RFI and the location of their sources.

## Law enforcement and public order

Law enforcement and other authorities are increasingly dealing with problems stemming from radio-controlled or radio-connected systems.

**Illegal drone use:**  These include people flying radio-controlled unmanned aerial vehicles (drones) in prohibited places. SCOS systems can be used to detect characteristic transmissions of drone operation in areas such as in the airfield flight traffic area.

**Detecting jamming devices:** A problem area for security staff and law enforcement is the use of radio jammers to interfere with remote control devices like vehicle keyless entry systems or radio links for alarm systems. SCOS devices can be used to identify and locate jamming systems.

**Detecting unauthorized mobile phone use:** Controlled and high security areas such as prisons will frequently prohibit the use of cellular phones in certain areas, but may not jam operating frequencies because of other regulations. Identifying and locating transmissions allows direct action to be taken on equipment users.

## Network Operator Applications

Radio planning for fixed radio deployment.

Spectrum forensics for identifying sources of interference.

# Informative: Metadata Specification

This appendix presents the structure of metadata used in the information message exchanged between Sensor Manager (TM) and Sensing Device (SD). First, metadata classification is reported in B.1. Reference source not found.. Sub-clause A.2 elaborates on data types and structures required in formally describing metadata. Sub-clause A.3 then formally describes these parameters exchanged based on the data type descriptions given by A.2.

## Metadata categorization

This sub-clause addresses the main categories of metadata included in the messages exchanged between SD and TM. They are categorized into three main classes having different purposes: Class A (System Metadata), Class B (Sensor settings) and Class C (Sensing related metadata).

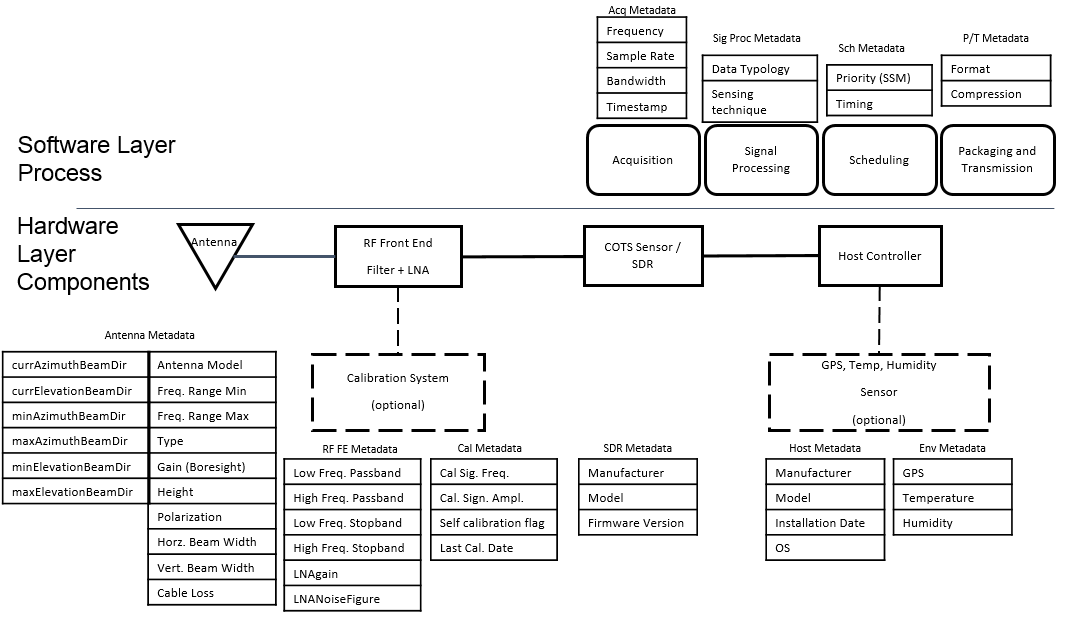


Figure 12 SD model - Hardware layer components and Software layer processes with relevant metadata

### System Metadata

Class A (System Metadata) includes all factory information related pieces of data and remain constant for the entire lifespan of the component (SD). Taking into account the metadata reported in figure A1, Antenna Metadata, RF Front-End Metadata, Calibration Metadata, SDR Metadata, Host Metadata are included in this category.

Class A metadata is not subjected to any change since it is offered as a response to a specific query in SD association process.

### Current Status Metadata

Class B (Current Status Metadata) includes data describing the actual configuration of the device, in terms of hardware (positioning, antenna configuration, etc.) and software (frequency settings, sampling rate, sensing algorithm, etc.).

Class B metadata is provided to TM, after a specific user request, and can be subjected to modification and special settings by the Tasking Agent.

### Sensing related Metadata

Class C (Sensing related metadata), specifying parameters strictly related to performed sensing action (scanned time, timestamp, atmosphere conditions, etc.);

Class C metadata is not subjected to any change since it is offered as a response to a specific query in a Sensing request.

## Data Types

This sub-clause defines the primitive data types, simple data types, and derived data types used in the definition of metadata defined in A.3. The physical units used in this definition are based on the International System of Units (SI) and are summarized in Table 1.

Table A1: Units used in the description of Metadata

|  |  |  |  |
| --- | --- | --- | --- |
| Unit | Unit symbol | Value | Note |
| second | s |  | SI unit |
| Hertz | Hz | 1 Hz = 1/s | SI derived unit |
| meter | m |  | SI unit |
| Watt | W | kg•m2/s3 | SI derived unit |
| Power ratio | dB | [dB]=10•log10(P1[W]/P2[W]) | dimensionless |
| Power ratio respect to 1 mW | dBm | [dBm]=10•log10(P1[W]/1 mW) | dimensionless |
| radian | rad | 1 rad = 180/π | SI derived unit, dimensionless |
| degree of arc | ° | 1° = π/180 rad | dimensionless |
| Power ratio respect to isotropic antenna | dBi | [dBi]=10•log10(P1[W]/Pisotropic[W]) | dimensionless |
| Kelvin | K |  | SI unit |

### Simple data type

In this sub-clause are summarized the simple data type considered in this document.

#### Boolean

A primitive logical data type having one of two values of “true” (1, nonzero) or “false” (0, zero).

#### Integer

A primitive integral data type representing natural numbers and their negatives. Note that common binary representations limit the number range due to machine word length restrictions. In this standard, the integer length is defined as 32-bit value.

#### Unsigned integer

A primitive data type representing non-negative integers.

#### Float

A primitive data type storing real numbers, usually as floating-point numbers. Floating-point number representations as defined in IEEE Std 754TM-2008 can be taken as an example.

#### Array

A simple data type storing a collection of data values of a specified type.

#### String

A simple data type storing a sequence of bytes or characters. A string is a special use of a one-dimensional array.

### Complex data types

This sub-clause summarizes complex data types such as structured types or types that rely on specific interpretation or restriction of the underlying simple type.

#### Enumeration

An enumeration is a listing of elements mapped to an index set consisting of natural numbers. That is, each element of the set is unambiguously represented by an ordinal value.

#### Cluster

A complex data type that aggregates a fixed set of labelled elements, possibly of different simple types, into a single element.

#### Fixed-point

Fixed-point numbers are rational numbers with a fixed length mantissa and a fixed exponent. In contrast to a floating-point representation, using a fixed length but variable exponent, the value range is limited by the mantissa length, but the resolution is constant over the value range. They can be realized by using an integer value in conjunction with an implicit multiplier.

#### Unsigned fixed-point

Unsigned fixed-point numbers are non-negative fixed-point numbers.

## Description of metadata

Metadata descriptions are given in a tabular form throughout A.3.1 through A.3.21. They consist of the metadata name and ID, a short textual description, and a type and size specification, if needed.

**⎯ Metadata name and ID**

Metadata name provides a unique identification of the parameter in human readable form, whereas the numerical ID is given to unambiguously identify the metadata in the process of information exchange between SCOS entities.

**⎯ Metadata type and size**

Metadata types are of one of the types defined by A.2. Some parameters may be further restricted in their value range or magnitude. The size field of the parameter description is supplementary information that is either a fixed value, determined by the number of elements contained, or variable if at least one of the elements contained is optional, or is of variable size itself. To avoid implementation dependent specifications for metadata, size is always given in terms of the underlying type.

For metadata based on array types, the size is given as the number of elements stored in the array or as “variable” if the size of the array is unspecified. Note that variable size arrays demand for an implicit array length value in information exchange.

For cluster types, the aggregated size depends on the implementation of the elements enclosed and thus is omitted in the parameter description. The implementation then will decide on the binary representation, encoding, and size in terms of bits or bytes and any tag or length values needed.

Table 2 provides a summary of metadata and categorizes them into each of the three classes introduced in A.1.

Table A2: Summary of metadata

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **ID** | **Metadata name** | **Sub-clause** | **Class A** | **Class B** | **Class C** |
| 001 | Antenna | A.3.1 | X |  |  |
| 002 | Curr.Azi.Beam.Dir. | A.3.2 |  | X |  |
| 003 | Curr.Elev.Beam.Dir. | A.3.3 |  | X |  |
| 004 | Front-End | A.3.4 | X |  |  |
| 005 | Cal | A.3.5 | X |  |  |
| 006 | SDR | A.3.6 | X |  |  |
| 007 | Host | A.3.7 | X |  |  |
| 008 | Env | A.3. |  |  | X |
| 009 | RGeolocation | A.3.9 |  |  | X |
| 010 | Frequency | A.3.10 |  | X |  |
| 011 | Sampl.Rate | A.3.11 |  | X |  |
| 012 | Bandwidth | A.3.12 |  | X |  |
| 013 | Timestamp | A.3.13 |  |  | X |
| 014 | Scan.Time | A.3.14 |  |  | X |
| 015 | Data.Typology | A.3.15 |  |  | X |
| 016 | Sens.Algorithm | A.3.16 |  | X |  |
| 017 | Priority | A.3.17 |  | X |  |
| 018 | Timing | A.3.18 |  | X |  |
| 019 | Compression | A.3.19 |  |  | X |
| 020 | Format | A.3.20 |  |  | X |
| 021 | SDName | A.3.21 | X |  |  |
| 022 | SCOSOperator | A.3.22 |  |  |  |
| 023 | SDMode | A.3.23 |  | X |  |
| 024 | SDType | A.3.24 | X |  |  |
| 025 | SDID | A.3.25 | X |  |  |
| 026 | SDCert | A.3.26 | X |  |  |

### Antenna

The Antenna metadata indicates specifications of the antenna as given by the manufacturer. The definition of each parameters and the related unit of measurements are reported in table A3.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Antenna | **Meas. Unit:** | -------- | **Data type:** | Cluster |
| **ID:** | 001 | **Size**: | Variable |  |  |
| **Desc.:** | List of the Antenna specifications according to manufacturer specifications. | | | | |
| .0 | Antenna.Model | Data type: | String |  |  |
| .1 | Antenna.Freq.min | Data type: | Unsigned fixed-point |  |  |
| .2 | Antenna.Freq.max | Data type: | Unsigned fixed-point |  |  |
| .3 | Antenna.Type | Data type: | String |  |  |
| .4 | Antenna.Gain | Data type: | Float |  |  |
| .5 | Antenna.Polarization | Data type: | Enumeration |  |  |
| .6 | Antenna.Height | Data type: | Float |  |  |
| .7 | Antenna.Horz.Beam.Width | Data type: | Fixed-point |  |  |
| .8 | Antenna.Vert.Beam.Width | Data type: | Fixed-point |  |  |
| .9 | Antenna.Min.Azi.Beam.Dir. | Data type: | Unsigned fixed-point |  |  |
| .10 | Antenna.Max.Azi.Beam.Dir. | Data type: | Unsigned fixed-point |  |  |
| .11 | Antenna.Min.Elev.Beam.Dir. | Data type: | Fixed-point |  |  |
| .12 | Antenna.Max.Elev.Beam.Dir. | Data type: | Fixed-point |  |  |
| .13 | Antenna.Cable.Loss | Data type: | Float |  |  |
| .14 | Reserved (for future use) |  |  |  |  |

Table A3: Antenna metadata parameters description

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Name** | **Content** | | | | **Meas. Unit:** |
| **Antenna.Model** | It contains a string with the model of the installed antenna | | | | ----- |
| **Antenna.Freq.min** | Min input frequency value | | | | Hz |
| **Antenna.Freq.max** | Max input frequency value | | | | Hz |
| **Antenna.Type** | Antenna type | | | | ----- |
| **Antenna.Gain** | Antenna gain | | | | dBi |
| **Antenna.Polarization** | Antenna polarization | | | | ----- |
| Enumerator  Enumerator  Enumerator  Enumerator  Enumerator | “VL”  “HL”  “LHC”  “RHC”  “Slant” | Value:  Value:  Value:  Value:  Value: | 0  1  2  3  4 |
| **Antenna.Height** | Antenna height | | | | m |
| **Antenna.Horz.Beam.Width** | Horizontal 3-dB beamwidth | | | | ° (degree) |
| **Antenna.Vert.Beam.Width** | Vertical 3-dB beamwidth | | | | ° (degree) |
| **Antenna.Min.Azi.Beam.Dir.** | minimum direction of main beam in azimuthal plane expressed in degrees from N | | | | ° (degree) |
| **Antenna.Max.Azi.Beam.Dir.** | maximum direction of main beam in azimuthal plane expressed in degrees from N | | | | ° (degree) |
| **Antenna.Min.Elev.Beam.Dir.** | minimum direction of main beam in elevation plane expressed in degrees from horizontal plane | | | | ° (degree) |
| **Antenna.Max.Elev.Beam.Dir.** | maximum direction of main beam in elevation plane expressed in degrees from horizontal plane | | | | ° (degree) |
| **Antenna.Cable.Loss** | Attenuation introduced by the cable that connects the antenna with the RF front-end | | | | dB |

### Current Azimuth Beam Direction

Current Azimuth Beam Direction indicates the current direction of the antenna main beam in azimuthal plane. It is expressed in degrees from N and it is included in a range that has Antenna.Min.Azi.Beam.Dir. as lower boundary and Antenna.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **Curr.Azi.Beam.Dir** | **Meas. Unit:** | **° (degree)** | **Data type:** | **Unsigned fixed-point** |
| **ID:** | 002 | **Size:** | 1 |  |  |
| **Desc.:** | It indicates the current direction of the antenna main beam in azimuthal plane. | | | | | |
|  | Range (min/resolution/max) | | Antenna.Min.Azi.Beam.Dir. | 1° | Antenna.Max.Azi.Beam.Dir. | |

### Current Elevation Beam Direction

Current elevation Beam Direction indicates the current direction of the antenna main beam in elevation plane. It is expressed in degrees from horizontal plane and it is included in a range that has Antenna.Min.Elev.Beam.Dir. as lower boundary and Antenna.Max.Elev.Beam.Dir. as upper boundary.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **Curr.Azi.Beam.Dir** | **Meas. Unit:** | **° (degree)** | **Data type:** | **Fixed-point** |
| **ID:** | 003 | **Size:** | 1 |  |  |
| **Desc.:** | It indicates the current direction of the antenna main beam in elevation plane. | | | | | |
|  | Range (min/resolution/max) | | Antenna.Min.Elev.Beam.Dir. | 1° | Antenna.Max.Elev.Beam.Dir. | |

### RF Front-End Metadata

The RF Front-End metadata indicates specifications of the RF front-end as given by the manufacturer. The definition of each parameters and the related unit of measurements are reported in table A4.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **Front-End** | **Meas. Unit:** | **--------** | **Data type:** | **Cluster** | |
| **ID:** | 004 | **Size:** | Variable |  |  | |
| **Desc.:** | List of the RF Front-End specifications according to manufacturer specifications. | | | | | | |
| .0 | Front-End.Low.Freq.Passband | | Data type: | | | Unsigned fixed-point | |
| .1 | Front-End.High.Freq.Passband | | Data type: | | | Unsigned fixed-point | |
| .2 | Front-End.Low.Freq.Stopband | | Data type: | | | Unsigned fixed-point | |
| .3 | Front-End.High.Freq.Stopband | | Data type: | | | Unsigned fixed-point | |
| .4 | Front-End.LNA-Gain | | Data type: | | | Float | |
| .5 | Front-End.LNA-NF | | Data type: | | | Float | |
| .6 | Reserved (for future use) | |  | | |  | |

Table A4: Front-End metadata parameters description

|  |  |  |
| --- | --- | --- |
| **Name** | **Content** | **Meas. Unit:** |
| Front-End.Low.Freq.Passband | Low passband frequency evaluated at -1 dB | Hz |
| Front-End.High.Freq.Passband | High passband frequency evaluated at -1 dB | Hz |
| Front-End.Low.Freq.Stopband | Low stopband frequency evaluated at -60 dB | Hz |
| Front-End.High.Freq.Stopband | High stopband frequency evaluated at -60 dB | Hz |
| Front-End.LNA-Gain | Low Noise Amplifier Gain | dB |
| Front-End.LNA-NF | Noise Figure of Low Noise Amplifier | dB |

### Calibration Metadata

Calibration metadata contains information about last calibration date of SD and, if it is able to execute self-calibration procedure, it also contains information about the reference signal generator built in. The definition of each parameters and the related unit of measurements are reported in table A5.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **Cal** | **Meas. Unit:** | **--------** | **Data type:** | **Cluster** | |
| **ID:** | 005 | **Size:** | Variable |  |  | |
| **Desc.:** | List of calibration parameters according to manufacturer specifications. | | | | | | |
| .0 | Cal.Last.Cal.Date | | Data type: | | | Unsigned integer | |
| .1 | Cal.Self.Cal.Flag | | Data type: | | | Boolean | |
| .2 | Cal.Sig.Freq | | Data type: | | | Unsigned fixed-point | |
| .3 | Cal.Sig.Ampl | | Data type: | | | Float | |
| .4 | Reserved (for future use) | |  | | |  | |

Table A5: Calibration metadata parameters description

|  |  |  |
| --- | --- | --- |
| **Name** | **Content** | **Meas. Unit:** |
| Cal.Last.Cal.Date | The time stamp of the last calibration. It is denoted as a basic reference time value. Seconds since midnight (UTC) of January 1, 1970 absolute time. | s |
| Cal.Self.Cal.Flag | This is set to “1” if the sensor performs a periodical self-calibration procedure. Otherwise it is set to “0” if the self-calibration is performed after a user request. | ----- |
| Cal.Sig.Freq | Frequency of the internal calibration source. If the SD is not equipped with an internal calibration source this parameter is set to the default value 0. | Hz |
| Cal.Sig.Ampl | Amplitude of the internal calibration source. If the SD is not equipped with an internal calibration source this parameter is set to the default value 0. | dB |

### A.3.6 SDR Metadata

SDR metadata contains information about SD computing hardware specifications as given by the manufacturer. The definition of each parameters and the related unit of measurements are reported in table A6.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **SDR** | **Meas. Unit:** | **--------** | **Data type:** | **Cluster** | |
| **ID:** | 006 | **Size:** | Variable |  |  | |
| **Desc.:** | List of SDR parameters according to manufacturer specifications. | | | | | | |
| .0 | SDR.Manufacturer | | Data type: | | | String | |
| .1 | SDR.Model | | Data type: | | | String | |
| .2 | SDR.Firmware | | Data type: | | | String | |
| .3 | Reserved (for future use) | |  | | |  | |

Table A6: SDR metadata parameters description

|  |  |  |
| --- | --- | --- |
| **Name** | **Content** | **Meas. Unit:** |
| SDR.Manufacturer | SD manufacturer | ----- |
| SDR.Model | SD model | ----- |
| SDR.Firmware | Current firmware version | ----- |

### Host Metadata

Host metadata contains information about the host controller of the SD. The host controller can be included in SD hardware or cab an external device that control and drives the sensor. The definition of each parameters and the related unit of measurements are reported in table A7.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **Host** | **Meas. Unit:** | **--------** | **Data type:** | **Cluster** | |
| **ID:** | 007 | **Size:** | Variable |  |  | |
| **Desc.:** | List of Host parameters according to manufacturer specifications and installation. | | | | | | |
| .0 | Host.Manufacturer | | Data type: | | | String | |
| .1 | Host.Model | | Data type: | | | String | |
| .2 | Host.OS | | Data type: | | | String | |
| .3 | Host.Inst.Date | | Data type: | | | Unsigned integer | |
| .4 | Reserved (for future use) | |  | | |  | |

Table A7: Host metadata parameters description

|  |  |  |
| --- | --- | --- |
| **Name** | **Content** | **Meas. Unit:** |
| Host.Manufacturer | Manufacturer of the host | ----- |
| Host.Model | Model of the host | ----- |
| Host.OS | Operating system installed on the host | ----- |
| Host.Inst.Date | The date when SD has been installed. It is denoted as a basic reference time value. Seconds since midnight (UTC) of January 1, 1970 absolute time. | s |

### Environmental Metadata

Environmental metadata reports information about the environmental conditions experienced during the measurement process. The definition of each parameters and the related unit of measurements are reported in table A8.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Env | **Meas. Unit:** | -------- | **Data type:** | Cluster | |
| **ID:** | 008 | **Size:** | Variable |  |  | |
| **Desc.:** | List of Environment parameters. | | | | | | |
| .0 | Env.Temperature | | Data type: | | | Float | |
| .1 | Env.Humidity | | Data type: | | | Float | |
| .2 | Reserved (for future use) | |  | | |  | |

Table A8: Environmental metadata parameters description

|  |  |  |
| --- | --- | --- |
| **Name** | **Content** | **Meas. Unit:** |
| **Env.Temperature** | Environmental temperature | K |
| **Env.Humidity** | Environmental humidity | g/m3 |

### Reference Geolocation

This metadata indicates an absolute position of SD. This position is denoted considering the WGS 84 reference coordinate system or its successors (see National Imagery and Mapping Agency [ref]). The definition of each parameters and the related unit of measurements are reported in table A8.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **RGeolocation** | **Meas. Unit:** | -------- | **Data type:** | Cluster | |
| **ID:** | 009 | **Size:** | 3 |  |  | |
| **Desc.:** | Basic reference geolocation parameter indicating the absolute geolocation of a SD based on the WGS 84 reference coordinate system or its successors. | | | | | | |
| .0 | RGeolocation.Elev | | Data type: | | | Integer | |
| .1 | RGeolocation.Lat | | Data type: | | | Fixed-point | |
| .2 | RGeolocation.Long | | Data type: | | | Fixed-point | |

**Table A9: Reference geolocation metadata parameters description**

|  |  |  |
| --- | --- | --- |
| **Name** | **Content** | **Meas. Unit:** |
| **RGeolocation.Elev** | Elevation is the altitude with respect to sea level. | m |
| **RGeolocation.Lat** | Latitude | ° (degree) |
| **RGeolocation.Long** | Longitude | ° (degree) |

### Frequency

Frequency denotes the center frequency of the channel where the SD is currently tuned. This parameter is given in Hz. It is realized as an unsigned fixed-point value assuming a resolution of 1 Hz and a maximum range of 0 to 232–1 Hz (i.e., the equivalent of a 32-bit mantissa)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **Frequency** | **Meas. Unit:** | Hz | **Data type:** | Unsigned fixed-point |
| **ID**: | 010 | **Size**: | 1 |  |  |
| **Desc.:** | It denotes the center frequency of the channel where the SD is currently tuned. | | | | | |
|  | Range (min/resolution/max) | | 0 Hz | 1 Hz | (232–1) Hz | |

### Sample rate

It is the current sampling rate adopted during the signal acquisition process. This parameter is given in Hz. It is realized as an unsigned fixed-point value assuming a resolution of 1 Hz and a maximum range of 0 to 232–1 Hz (i.e., the equivalent of a 32-bit mantissa).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **Samp.Rate** | **Meas. Unit:** | Hz | **Data type:** | Unsigned fixed-point |
| **ID:** | 011 | **Size:** | 1 |  |  |
| **Desc.:** | It denotes the current sampling rate adopted during the signal acquisition process. | | | | | |
|  | Range (min/resolution/max) | | 0 Hz | 1 Hz | (232–1) Hz | |

### Bandwidth

Bandwidth denotes the width of the channel where the SD is currently tuned. This parameter is given in Hz. It is realized as an unsigned fixed-point value assuming a resolution of 1 Hz and a maximum range of 0 to 232–1 Hz (i.e., the equivalent of a 32-bit mantissa).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Samp.Rate | **Meas. Unit:** | Hz | **Data type:** | Unsigned fixed-point |
| **ID:** | 011 | **Size:** | 1 |  |  |
| **Desc.:** | It denotes the current sampling rate adopted during the signal acquisition process. | | | | | |
|  | Range (min/resolution/max) | | 0 Hz | 1 Hz | (232–1) Hz | |

### Timestamp

Timestamp denotes the time instant in which the SD has completed the acquisition.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Timestamp | **Meas. Unit:** | s, s | **Data type:** | Cluster | |
| **ID**: | 013 | **Size:** | 2 |  |  | |
| **Desc.:** | It is the time instant in which the SD has completed the acquisition. It is denoted as a basic reference time value. Seconds since midnight (UTC) of January 1, 1970. | | | | | | |
| .0 | Timestamp.s | | Data type: | | | Unsigned integer | |
| .1 | Timestamp.us | | Data type: | | | Unsigned integer | |

### Scanned Time

This metadata denotes the amount of time that the SD has employed to perform the sensing operation in a channel or set of channels.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Scan.Time | **Meas. Unit:** | ms | **Data type:** | Unsigned fixed-point |
| **ID:** | 014 | **Size:** | 1 |  |  |
| **Desc.:** | It denotes the amount of time that the SD has employed to perform the sensing operation in a channel or set of channels. | | | | | |
|  | Range (min/resolution/max) | | 0 ms | 1 ms | (232–1) ms | |

### Data Typology

This metadata denotes the description of the received data domain.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Data.Typology | **Meas. Unit:** | -------- | **Data type:** | Enumeration | |
| **ID:** | 015 | **Size:** | 1 |  |  | |
| **Desc.:** | Unique identification of the possible data typology of the received data. | | | | | | |
| Enumerator | I/Q.Time | | Value: | | | 0 | |
| Enumerator | FFT | | Value: | | | 1 | |
| Enumerator | PSD | | Value: | | | 2 | |
| Enumerator | I/Q.Freq | | Value: | | | 3 | |
| Enumerator | Time.sample | | Value: | | | 4 | |
| Enumerator | Occupied | | Value: | | | 5 | |
| Note: FFT stands for Fast Fourier Transform and PSD stands for Power Spectral Density. I/Q.Time and I/Q.Freq denote the I/Q sample in time domain and the spectrum of the I/Q component, respectively. Time.sample is the acquired signal and occupied is used if the output is Boolean variable (the SD has scanned only one channel) or an array of Boolean variables (the SD has scanned more than one channel). “1” means that the scanned channel is occupied “0” otherwise. | | | | | | | |

### Sensing Technique

It denotes the sensing technique currently adopted by the SD.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Sens.Algorithm | **Meas. Unit:** | -------- | **Data type:** | Enumeration | |
| **ID:** | 016 | **Size:** | 1 |  |  | |
| **Desc.:** | Unique identification of the sensing technique currently adopted by the SD. | | | | | | |
| Enumerator | Cyclostationarity | | Value: | | | 0 | |
| Enumerator | Energy.detection | | Value: | | | 1 | |
| Enumerator | Custom | | Value: | | | 2 | |

### Priority

Priority denotes the scheduling scheme used for incoming request.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Priority | **Meas. Unit:** | -------- | **Data type:** | Enumeration | |
| **ID:** | 017 | **Size:** | 1 |  |  | |
| **Desc.:** | Unique identification of the priority criterium currently employed by the SD. | | | | | | |
| Enumerator | FCFS | | Value: | | | 0 | |
| Enumerator | RR | | Value: | | | 1 | |
| Enumerator | Custom | | Value: | | | 2 | |
| Note: FCFS stands for First Come First Served and RR stands for Round Robin. | | | | | | | |

### Timing

Timing metadata denotes how the SD currently manages the sensing request that it receives.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Timing | **Meas. Unit:** | -------- | **Data type:** | Enumeration | |
| **ID:** | 018 | **Size:** | 1 |  |  | |
| Desc.: | Unique identification of the timing criterium currently employed by the SD. | | | | | | |
| Enumerator | On.Demand | | Value: | | | 0 | |
| Enumerator | Timed.with.periodicity | | Value: | | | 1 | |

### Compression

Compression metadata denotes the compression algorithm used in order to reduce the amount of data to transmit.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Compression | **Meas. Unit:** | -------- | **Data type:** | Enumeration | |
| **ID:** | 019 | **Size:** | 1 |  |  | |
| **Desc.:** | Unique identification of the compression algorithm used by the SD in order to reduce the amount of data to transmit. | | | | | | |
| Enumerator | LZ | | Value: | | | 0 | |
| Enumerator | DEFLATE | | Value: | | | 1 | |
| Note: LZ is the Lempel-Ziv compression method. | | | | | | | |

### Format

Format metadata denotes data transmission format used for data transmission by the SD.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | Format | **Meas. Unit:** | -------- | **Data type:** | Enumeration | |
| **ID**: | 020 | **Size:** | 1 |  |  | |
| Desc.: | Unique identification of the data transmission format used for data transmission by the SD. | | | | | | |
| Enumerator | Little.Endian | | Value: | | | 0 | |
| Enumerator | Big.Endian | | Value: | | | 1 | |

### Sensing device name

This metadata denotes the name of the sensing device registered with SCOS operator.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **SDName** | **Meas. Unit:** | ----- | **Data type:** | String |
| **ID:** | 021 | **Size:** | 1 |  |  |
| Desc.: | It denotes the name of the sensing device registered with SCOS operator. | | | | | |

### SCOS operator

It denotes the name of the SCOS operator.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | **SCOSOperator** | **Meas. Unit:** | ----- | **Data type:** | String |
| **ID:** | 022 | Size: | 1 |  |  |
| Desc.: | It denotes the name of the SCOS operator. | | | | | |

### SD Mode

This metadata indicates the mode in which the SD is currently operating.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | SDMode | **Meas. Unit:** | -------- | **Data type:** | Enumeration | |
| **ID:** | 023 | **Size:** | 1 |  |  | |
| Desc.: | Unique identification of the mode in which the SD is currently operating. | | | | | | |
| Enumerator | Online | | Value: | | | 0 | |
| Enumerator | Offline | | Value: | | | 1 | |

### Type of the sensing device

This metadata indicates the type of SD.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | SDType | **Meas. Unit:** | -------- | **Data type:** | Enumeration | |
| **ID**: | 024 | Size: | 1 |  |  | |
| Desc.: | Unique identification of the SD type. | | | | | | |
| Enumerator | SD.Full | | Value: | | | 0 | |
| Enumerator | SD.Proxy | | Value: | | | 1 | |

### Sensing Device ID

This metadata indicates the unique ID associated to the SD.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Metadata name:** | SDID | **Meas. Unit:** | -------- | **Data type:** | Unsigned integer |
| ID: | 025 | Size: | 1 |  |  |
| Desc.: | The unique ID assigned to the sensing device. | | | | | |

### SD certificate file

This metadata gives information about certificate file of the considered SD.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Metadata name:** | SDCert | **Meas. Unit:** | ----- | **Data type:** | Cluster | |
| **ID:** | 026 | **Size**: | 3 |  |  | |
| Desc.: | It gives information about certificate file of the considered SD. | | | | | | |
| .0 | SDCert.Path | | Data type: | | | String | |
| .1 | SDCert.Name | | Data type: | | | String | |
| .2 | SDCert.Cert.Auth | | Data type: | | | String | |

### SD Task Control metadata

#### Scheduler Specification

|  |  |  |
| --- | --- | --- |
| **Algorithm** | **Value** | **Notes** |
| Unspecified | 0 |  |
| Host Controller | 1 |  |
| Embedded Job Controller | 2 |  |
| Multilevel | 3 |  |

#### SD Output Specification

|  |  |  |
| --- | --- | --- |
| **Algorithm** | **Value** | **Notes** |
| Unspecified | 0 | Invalid |
| Time domain IQ | 1 | Default |
| Freq. domain IQ | 2 |  |
| Time domain Amp, Phase | 3 |  |
| Freq. domain Amp, Phase | 4 |  |

### SCOS Association Metadata

Following table enumerates sensing device metadata for associating with SCOS.

|  |  |  |
| --- | --- | --- |
| Parameter | R/O/C | Description |
| NAME: SDName DATATYPE: string | Required | The name of the sensing device registered with SCOS operator.  The maximum length is 64 octets. |
| NAME: SCOSOperator DATA TYPE: string | Required | The name of the SCOS operator.  The maximum length is 64 octets. |
| NAME: SDMode DATA TYPE: Integer | Required | The mode in which sensing device operates. (1=online, 2=offline) |
| NAME: SDType DATA TYPE: Integer | Required | The type of the sensing device. (1=SDFull, 2=SDProxy) |
| NAME: SDID DATA TYPE: string | Required | The unique ID assigned to the sensing device. The maximum length is 64 octets. |
| NAME: SDCertFile  DATA TYPE: String | Required | The path of the SD certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: SDKeyFile  DATA TYPE: String | Required | The name of the SD certificate file.  The maximum length of the ID string is 256 octets. |
| NAME: SDCAFile  DATA TYPE: String | Required | The name of the trusted certificate authority.  The maximum length of the ID string is 256 octets. |

# Normative Functional Requirements

## Data Client Requirement

Data Clients accessing a distributed SCOS system will require access to one or more spectrum sensor devices situated at a remote location through an Internet-connected interface.

This interface must expose all of these functions to the SCOS system user through a defined, standardized interface

* The DCli can discover the availability and capability of sensor devices connected to a Sensor Management System
* The DCli can request the performance of a scan task according to chosen parameters, or in a more advanced system request a recurring scan to be scheduled that will be executed automatically by the SCOS system
* The DCli can define where the data generated in the scan should be transmitted to once complete
* The DCli must be given diagnostic or performance information relating to the execution of their scan task

The user must be able to access their scan data and associated metadata describing the scanner’s environment, hardware and software configuration and scan settings.

## Data Quality and Definition

Data acquired must be accompanied by adequate metadata information to allow for duplication of the measurement or assessment of data quality by a subject matter expert. Hardware specification and measurement parameters are to be included in the messaging metadata requirements. Information less amenable to metadata capture should be made available via appropriate and accessible documentation, e.g., algorithm described via written article with source code in Github repository.

## Regulatory requirements

This standard should provide mechanisms to meet the regulatory requirements of national operators that have defined parameters or requirements for spectrum sensing in various applications. These regulatory requirements would take two forms: the first is technical requirements for sensitivity, resolution, etc. The second is limitations on how and where sensing might be done where there are sensitivities around privacy, military use and other national policies and regulations.

## Policy Management and Enforcement Requirements

To allow for granularity in what the SCOS systems can do, but also ensure spectrum occupancy or utilization data is not exposed in contravention of national policy or regulation, it is proposed that the TM would be able to apply policies to allow or disallow certain functionality in the SDs, or disallow transmission of the data to third party systems.

These policies would be determined by the TM operator in accordance with their requirements and that of local authorities (e.g. a national regulator or network operator), and cascaded down to any connected SDs. These policies would allow sensing only according to allowed metrics (e.g. no hi-resolution raw scans in military radar bands), and limit sensing data transmission to certain classes of third party systems.

## Sensor Location-Fixing Requirements

The SCOS device can convey the location of the sensors to the aggregation entity such as the TM. The instruction to use available location capabilities on the SD (e.g. GPS location) will be part of the scan schedule instruction from the Data Client requiring the scan. This feature allows the TM or the aggregation entity to localize the proximity of the signal source location allowing more efficient spectrum management. This location fixing capability will be implemented by the system operator to be in accordance with local regulatory requirements.

Location can be fixed in four ways:

* At scan-time from lat/long co-ordinates from internal GPS, GLONASS or similar system
* Configured at startup-time from lat/long co-ordinates from internal GPS, GLONASS or similar system
* Configured on SD by authorised/certified installer at commissioning time with accurate lat/long coordinates
* Configured on SD by authorised/certified installer at commissioning time with street address/location

The type of location fix is specified in device metadata

## Service Level Agreement Requirements

*To be completed (contributions needed).*

## Certification Requirements

*To be completed (contributions needed).*

## Technical Requirements

### Device classes and complexity

The following sensing device categories may be considered:

* **Energy Efficient Sensing Devices**: This standard should provide mechanisms of energy efficient operations, eg. solar powered or battery operated spectrum sensors for monitoring applications.
* **Small form factor devices**: Devices that can fit the spectrum sensing function within a small form factor (e. g. a USB dongle, cell phone etc.)
* **Advanced Spectrum Sensing Devices**: Advanced Spectrum Sensing Devices with capable Radio Frequency Front Ends (RFFE) and dedicated resources for spectrum sensing may be considered.
* **Non-dedicated Devices with Sensing Capability:** A number of consumer and professional radio devices contain radio receivers that can be used as sensing devices, including mobile phone handsets, Wi-Fi access points (from 802.11ac) and Dynamic Spectrum Access radio systems (including 802.22).

### Number of devices

This standard shall support at least one Spectrum Sensing Device to cover a location or area, communicating with a back-end Spectrum Sensing Management System (TM), but will extend to describing an architecture and interfaces for multiple SDs potentially communicating with multiple TM instances.

### Real-time applications

The sensing devices will be performing spectrum sensing functions according to its scheduler (which is managed by its TM), which can be updated in near-real time (dependent on speed of communication between Data Client, TM and SD), or perform scans at scheduled intervals based on pre-configured schedules. However, the spectrum sensing reporting of data is out on a Best Effort basis, since the SCOS System uses the chosen available transport mechanism (e. g. 802.11, 802.22, Ethernet, Cable, Cellular etc.).

The SCOS system will benefit if sensing reports from various sensors are provided on a reasonable time-scales (e. g. minutes) so that the information is not stale. However, this is not a mandatory requirement. Also, the messaging format may be defined such that it does not produce excessive overhead penalty on the transport layer being used.

It is envisioned that real-time streaming will be provided for in future drafts of 802.22.3.

### Channelization

This standard may specify a Spectrum Manager entity that can command various sensors to go and sense in certain bands, or it may even specify the spectrum sensors to ignore certain bands from sensing, and impose channelization maps for sensors to meet local regulations or technical requirements.

Specific channelization maps may be provided in future drafts of 802.22.3.

## Security Requirements

The standard mandates secure authentication and authorisation between Data Client and TM, SD and TM, TM and Data Manager, and Data Manager and DC. Traffic between these components must also be encrypted. The specific security technology to be used is not mandated in this standard, but recommended best practices are described in Annex B. Note that the security model does not extend past transmission to the DC. Responsibility for securing the spectrum data at destination remains the responsibility of the operator of that store.

The technology model is designed to ensure that data derived from SCOS devices and TM are not used as an attack vector against White Space Databases, regulator spectrum management databases, etc. It is also designed to ensure that only authorised Data Clients can make use of the SCOS resources, and that they are correctly identified to enable correct application of the relevant system policies.

In each case, the security model must address:

(1) Data categorization (i.e. sensitivity/confidentiality of scan data)

(2) Access control - authorization and authentication (of each element when interacting with another)

(3) Logging and auditing (of instructions, tasks, access control)

(4) Data encryption (within devices and in transmission)

(5) System and information integrity (validation of device configuration, storage system)

*(NOTE: Section to be developed further. Contribution needed)*

### Intra-device Layer Security (physical interfaces)

This standard defines security mechanisms to ensure integrity of sensing chain from antenna to DC.

* Antenna to amplifier/filters: physical security of device in terms of cable/connectors (tampering such as substituting antenna, physical such as connector corrosion)
* Amplifier to SDR: cable connectors or PCB connections
* SDR to processing unit: cable connectors or PCB connections
* Enclosure for active elements: Protection against moisture, dust ingress. Screening against RFI from external sources. Screening to protect antenna elements against RFI from active elements.

*(NOTE: Section to be developed further. Contribution needed)*

### Inter-Layer Security

#### Network Layer

Since this standard uses any available transport mechanism for data transmission, it will not recommend its own security mechanisms, but will use the existing security mechanisms of the transport mechanism being used (e.g. network 802.11 using Transport Layer Security,

#### Application Layer

Data transmissions should be secured on the application layer using mechanisms to guarantee the integrity and confidentiality of sensing and control data transmissions. This standard does not specify the technology used, but recommended implementation practices are noted in Annex B: Device and System Security Recommendations.

### Security of sensed data

This Standard shall not support mechanisms that expose data of radio system users that are modulated onto signals that are examined by the 802.22.3 SCOS System. For example, any kind of demodulation of the signals that may interfere with the privacy of the radio system users shall not be not be supported. However, the SCOS system shall support sophisticated spectrum sensing methods such as cyclostationary processing that can detect signals and characterize their modulation type.

### Security of analyzed or characterized spectrum data

This Standard shall support security mechanisms to ensure that spectrum characterization data is transmitted to the destination DC in a secure way. This Standard shall not specify the security mechanisms used to protect this data once received by the DC – this is implementation and system user specific.

# SCOS Operational Modes

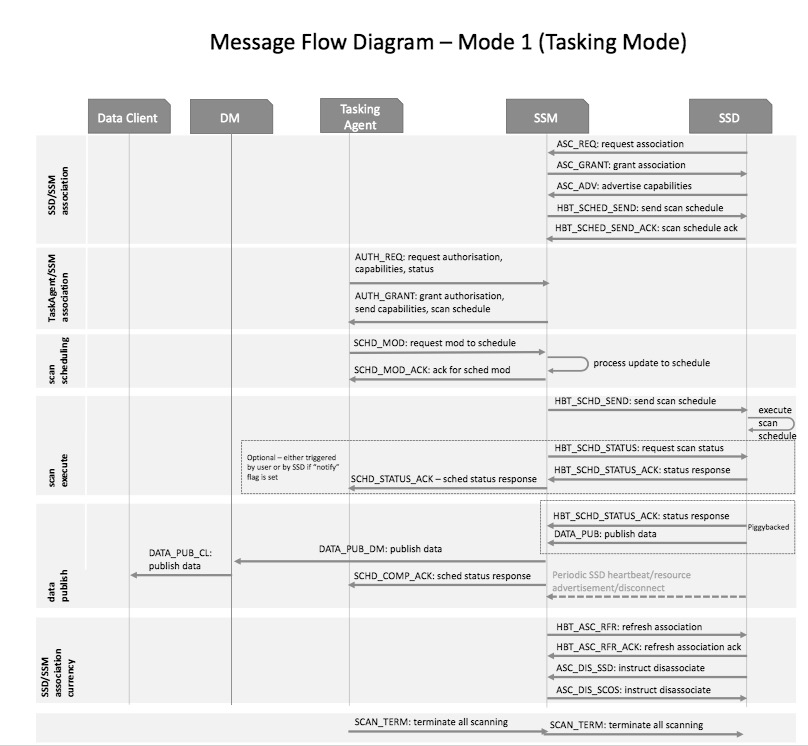
To allow great system flexibility with ability to meet multiple unknown use cases, but also allow a simplified task-specific operational use, two Operational Models are proposed:

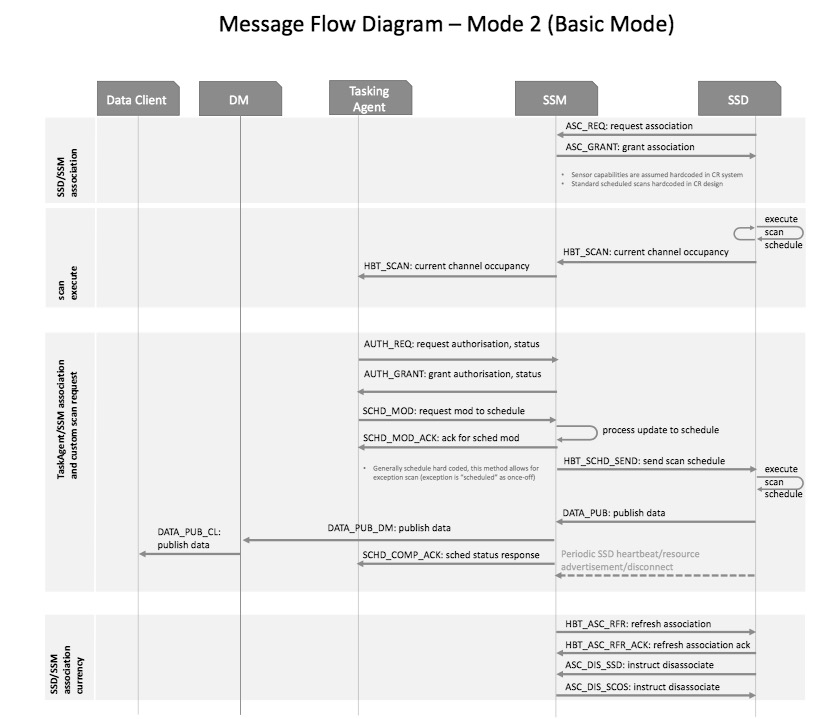
Mode 1 (full mode): This is a full-featured mode suitable for wide application, where the TM acts as a management device to allow multiple different users (“Data Clients”) to do different scans

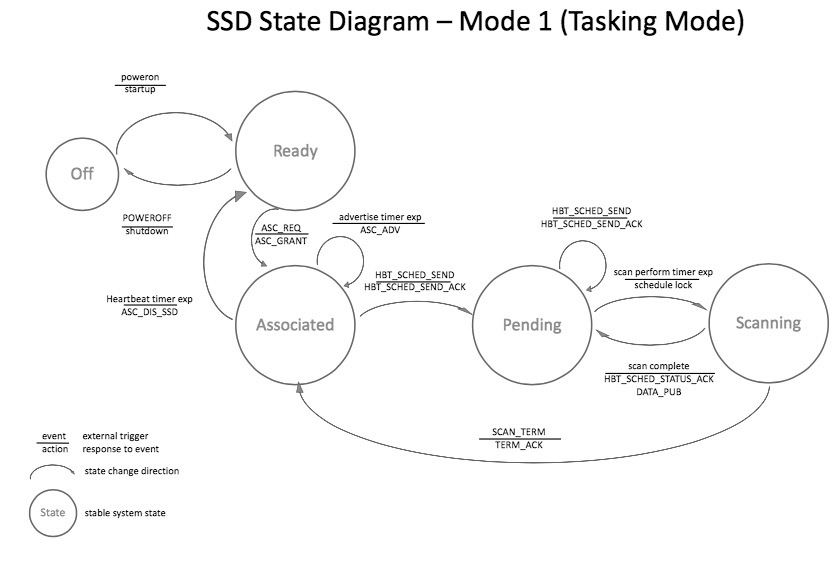
Mode 2 (minimal mode): This mode is suitable for cognitive radio implementations, where a sensing device is used in a semi-fixed configuration, reporting channel occupancy to the radio management system over heartbeat messages for low overhead, with some capability to perform specific scans as a task to let a radio supervisor system request a detailed scan.

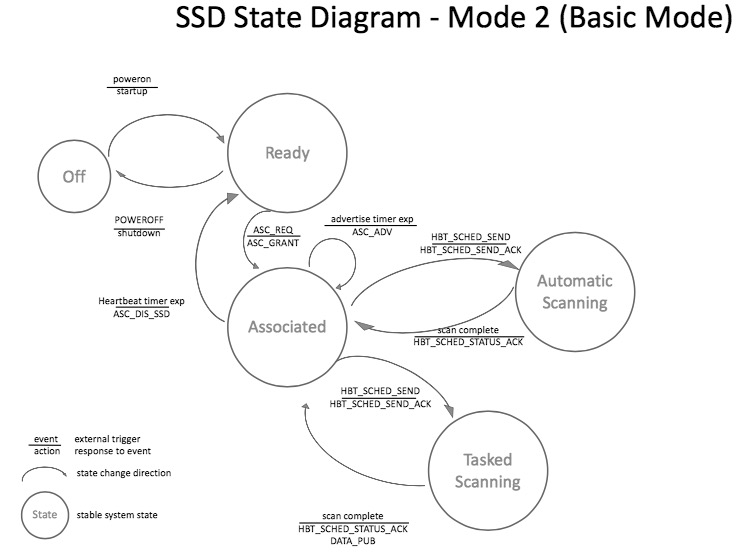
Mode 2 is a subset of Mode 1, using the same interfaces, primitives and protocols.

Further, Mode 3 (Offline Mode) is proposed for further examination and inclusion in later versions of standard. This mode would enable sensing devices to be given a task schedule, and then operate offline from the SCOS management systems, and synchronize data and tasks later when re-associated to management systems.









# SCOS Topology Examples

An SCOS system consists of *a single TM, a single DM* which communicate over any standard network transport with one or more SDs, DClis, and DCs.

Figure 13: SCOS System Topologyillustrates the above described SCOS system topology.

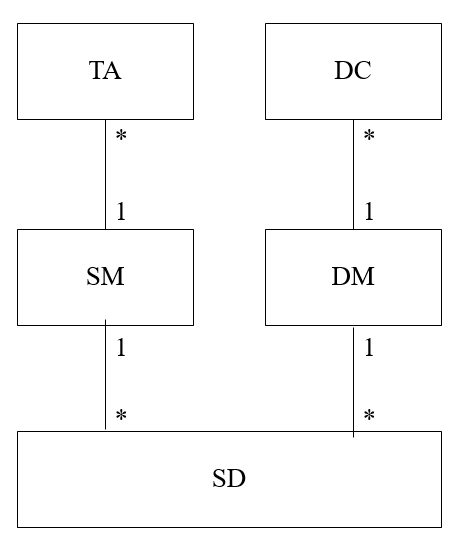


Figure 13: SCOS System Topology

## Non 802.22.3 compliant SDs and SCOS Cascading

The 802.22.3 SCOS standard makes provision for proxying, that allows a non 802.22.3 compliant SD to associate with and be controlled by an TM, as well cascading of systems, where one 802.22.3 compliant TM to be associated with, and delegate tasks to, another 802.22.3 SCOS system.

SD Proxy facilitates an TM communicate with proprietary sensing hardware, acting as a software translation mechanism that translates between SCOS messages.

TM Proxy enables cascading of SCOS systems where an TM can communicate with other SMs as if they were associated SDs.

Figure 14 illustrates the extensions with an instance of system topology.

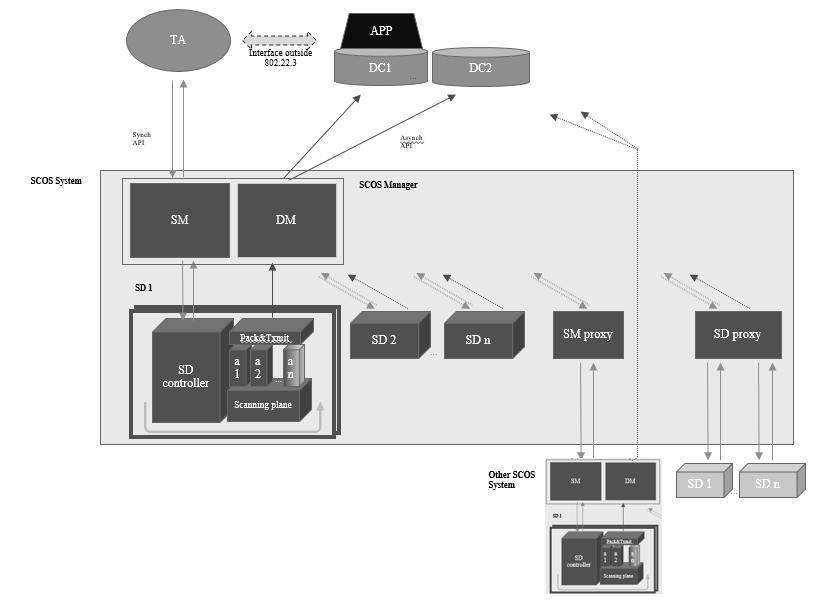


Figure 14: An SCOS System Instance with Extensions

# System Policy Model

## SCOS Policy

A policy layer in the TM at the northbound and southbound API will ensure that the TM is operated within requirements of a local authority (national regulator, law enforcement, military, etc).

The policy on the southbound interface will determine, based on the USER type, (e.g. how a central authority can define what kinds of sensing can be done in what bands, what data governance rules there are, etc

-- resource allocation – what kinds of users are authorized to request resources from the sensor network and in which priority (i.e. if a sensing network is resource constrained, who gets first dibs on the sensors)

Policy on the northbound interface (TM>DBstore) will have rules for how sensing data may be distibuted, and data storage policies. This would include which scan data takes priority if local storage in the store & forward buffer is running out due to a failed transmission link, and how long certain USER data is allowed to be stored in the local store & forward buffer.

Based on SCOS platform architecture, the SCOS policy is categorized based into following categories:

* Spectrum Task Manager (TM) Policy
* Spectrum Sensing Device (SD) Policy
* Sensing Data Manager (DM) Policy
* SCOS Platform Administration (SPA) Policy
* SCOS Platform User (SPU) Policy

The SCOS policy is expressed using JSON. Following subsection provides details about the schema of SCOS policy.

### The SCOS policy schema:

Each SCOS policy is associated with a policy-name, policy-namespace, policy-category, policy-scope, optional policy-description, and one or more statement(s).

Following figure shows the structure of a SCOS policy.

**SCOS Policy**

Top-level attributes

Statement

Statement

**Statement**

StatementId

Policy-Action

Actor

Resource

SCOS-Task

Condition-Block

Figure 15 High-level structure of the SCOS Policy

Example:

{

# "Version": "2017-02-15",

"Policy": {

"namespace": "OperatorFoo",

"name": "Calibration-access-control",

"description": "This policy added by FooAdmin On this date."

"Policy-type": "SPA-Policy",

"Policy-Action": "permit",

"Resource": "Foo:Sensors::\*"

"SCOS-Task": "Calibration-operation"

"Scope": "Sensor-management:"

"Condition": {

"equals" : {

"Actor" : "FooAdminRole"

}

}

}

The default namespace is global. Specific-Namespaces could be used to restrict the application of policy within a certain context. Namespaces avoids name collisions and enables to identify actors or resources uniquely (when the names have been reused across namespaces).

**SCOS Policy Statement**

Each statement specifies certain action. Different categories of SCOS-Policies are associated with different actions.

A statement may also have optional attributes that identify the context of applying the policy. These attributes allow specifying a fine-grained policy. Example context-attributes are Actor, Resource, SCOS-Task, time, frequency, and location.

A statement has an optional condition-block. The action is performed only when the condition(s) are matched.

**Data Client**

A Data Client is an entity that wishes to use the SCOS platform. A Taskin Agent could be specified in terms of role, user, or a user-group.

**User**

An user is an individual Data Client with specified name and access-credentials.

**User-group**

A user-group is a logical collection of users. A user-group is specified with name and access-credentials.

**Role**

A role is specified with a name. The role could be associated with specific SCOS services/functionality. A role could also be associated with privilege of various users of SCOS platform.

An user (or a user-group) is associated a role.

**Resource**

SD and Sensing data are two prime resources within SCOS platform.

**Resource-group**

Multiple SDs could be grouped together to jointly specify policies for using the SDs. SDs could possibly be grouped based on various attributes such as location, SD-hardware-type, SD-software-type.

**Namespace**

Actors or resources are associated with a namespace. This avoids name collisions and enables to identify actors or resources uniquely (when the names have been reused across namespaces).

**SCOS-Task**

An SCOS-task represents a specific sensing-task within SCOS platform issued by a particular Actor on particular resources. Additionally, a policy may specify pre-defined SCOS operations in the SCOS-Task. These predefined SCOS-Tasks include: Scanning, Calibration, Storage, and Transmission.

**Task-group**

Multiple tasks could be grouped together for convenience in specifying policies. For example, various tasks that can be performed towards sensor-management for a particular SD operator could be grouped together and referred to in the SCOS policy. Similarly, sensing data management related tasks could be grouped together for precisely and conveniently specifying sensing-data-management related policies.

**Conditions**

A condition is specified with a triplet of field(key), conditional-operator, and value. Condition is optional within a statement.

A condition evaluates whether a field meets certain criteria. Following table identifies various conditional operators.

|  |  |
| --- | --- |
| **Conditional-operator Name** | **Syntax** |
| equals | "equals" : "<value>" |
| Like | "like" : "<value>" |
| Contains | "contains" : "<value>" |
| In | "in" : [ "<value1>","<value2>" ] |
| Exists | "exists" : "<bool>" |
| LessThan | "lessthan" : "<value>" |
| GreaterThan | "greaterthan" : "<value>" |
| LessThanEquals | "lessthanequals" : "<value>" |
| GreaterThanEquals | "greaterthanequals" : "<value>" |

**Logical Operators**

Logical operators enable to manipulate or combine multiple conditions. Following table specifies the logical operators.

|  |  |
| --- | --- |
| **Logical operator** | **Syntax** |
| Not | “not”: {<condition>} |
| AllOf | "allOf" : [ {<condition>},{<condition>}] |
| AnyOf | "anyOf" : [ {<condition>},{<condition>}] |

**Aliases**

Aliases add convenience. Using aliases, multiple users can be combined together or multiple resources can be combined together to be referred in the SCOS policy . Furthermore, multiple tasks can be combined using task-groups.

Furthermore, locations could be specified using aliases to capture latitude, longitude, and altitude. A group of frequencies could also be combined using aliases. A group of time-slots also could be combined using aliases.

#### TM Policy Schema

Each TM policy has following required fields: PolicyName, PolicyScope, PolicyType, and PolicyAction.

Optional fields include: Policy-Description, condition-block, Actor, Resource, and SCOS-Task.

With Policy-Action ‘set’, TM attribute and value(s) could be specified.

#### SD Policy Schema

Each SD policy has following required fields: PolicyName, PolicyScope, PolicyType, PolicyAction, and Resource.

Optional fields include: Policy-Description, condition-block, Actor and SCOS-Task.

Policy-Actions: Set, Permit, Deny, Calibrate, Scan,

With Policy-Action ‘set’, SD attribute and value(s) could be specified.

#### DM Policy Schema

Each DM policy has following required fields: PolicyName, PolicyScope, PolicyType, and PolicyAction, and SCOS-Task.

Optional fields include: Policy-Description, condition-block, Actor and Resource.

Policy-Actions: Set, Permit, Deny, Transmit-Sensing-Data, Store-Sensing-data, Discard-Sensing-data

With Policy-Action ‘set’, DM attribute and value(s) could be specified.

#### SPU Policy Schema

Each SPU policy has following required fields: PolicyName, PolicyScope, PolicyType, PolicyAction, and Actor.

Optional fields include: Policy-Description, condition-block, SCOS-Task, and Resource.

Policy-Actions: Set, Permit, Deny

With Policy-Action ‘set’, SPU attribute and value(s) could be specified.

#### SPA Policy Schema

Each SPA policy has following required fields: PolicyName, PolicyScope, PolicyType, and PolicyAction.

Optional fields include: Policy-Description, condition-block, SCOS-Task, Actor, and Resource.

Policy-Actions: Set, Permit, Deny,

With Policy-Action ‘set’, SPA attribute and value(s) could be specified.

#### Policy Specification

In the first version of this standard that the TM and DM store a policy file which is installed manually by the SCOS operator (through mechanism such as SSH and update pull, or remote SCP).

### Policy Evaluation

Whenever an SCOS API needs to be executed, TM needs to confirm if the action is permitted by evaluating related policies.

There exist three scopes for SCOS policies: Sensing management scope, Sensing-data management scope, and Sensor-management scope. Depending on the API, policies in the appropriate scope are looked up.

The second step is ensure that the actor is authorized to perform tasks on the resource. A specific accept policy or default-accept policy should be match for the user, user-group, or role.

The final step is ensure if the resource permits the intended task. A specific accept policy or default-accept policy should be match for the resource, or resource-group.

### SCOS Policy Examples

#### SD Policy

Set sensitivity to -114 dBm task frequency UHFBand

PolicyID: <generated>

PolicyName: SCOSMinSensitivityRule

Policy-Category: SD-Policy

PolicyDescription: It applies to all SDs within the SCOS operational region.

Policy-Action: Set

Sensitivity: Value

Frequency: value

Discuss: Should frequency be within the context-block or condition-block?

#### TM Policy

Set scheduling minimum slot duration

PolicyID: <generated>

PolicyName: SSMMinSensingSlotDuration

Policy-Category: TM-Policy

PolicyDescription: It specifies the minimum slot duration for sensing task. The value is in seconds.

Policy-Action: Set

Min-sensing-slot-duration: Value (seconds)

Note: Fine-grained policy could be specified for a particular resource (SDs) or sensing-tasks.

Set sensing behavior for prioritized scan

PolicyID: <generated>

PolicyName: TM-Prioritized-Scan-Behavior

Policy-Category: TM-Policy

PolicyDescription: It specifies whether existing scan should be suspended if a prioritized scan-request is received.

Policy-Action: Set

Prioritized-scan-enabled: true

Condition-block: if wait-time greaterthan value (in seconds)

Note: Condition-block is optional. Condition-block can be used to specify a condition when existing scans can be suspended.

#### DM Policy

Set max-data-storage-duration at DM

PolicyID: <generated>

PolicyName: SDMMaxStorageConfig

Policy-Category: DM-Policy

PolicyDescription: It specifies how long DM can hold the sensing data.

Policy-Action: Set

Max-data-storage-duration: Value (seconds)

Note: Optionally specify task or SDs or SDS. The value is in seconds.

Discard sensing-data for <scan-task-L-band-User-Jim> if sensing-data is unqualified.

PolicyID: <generated>

PolicyName: User-Jim-L-Band-Discard-Data-Policy

PolicyDescription: If sensing data does not meet the criteria specified in the sensing task, discard the data.

Policy-Category: DM-Policy

Policy-Action: discard-data

SCOS-Task: scan-task-L-band-User-Jim

Condition-block: sensing-data-quality is ‘unqualified’.

Note: The condition-block identifies when the operation is performed. Here, sensing-data has attribute sensing-data-quality. The condition is satisfied when the attribute’s value is unqualified. The sensing-task is identified pre-defined using name-alias. Optionally, the policy could be made more specific for certain time, and location attributes.

#### SPA Policy

Enable TM-Proxy device usage in SCOS system.

PolicyID: <generated>

PolicyName: Enable-SD-Proxy-Config

Policy-Category: SPA-Policy

PolicyDescription: Enable TM-Proxy devices in the SCOS platform.

Policy-Action: Set

TM-Proxy-Enabled: Boolean-Value

Note: Optionally, the policy could be made more specific for certain frequency, time, and location attributes.

#### SPU Policy

Deny scan operation for User-Foo in the military bands

PolicyID: <generated>

PolicyName: MilitaryBandScanRestrictionPolicy

Policy-Category: SPU-Policy

PolicyDescription: Deny certain users/roles/groups to scan in certain bands.

SCOS-Task: scan

Actor: User-Foo

Policy-Action: Deny

frequency: X-band

Note: The actors could be specified with user/role/user-group. The frequency bands could be pre-defined using name-aliases. Optionally, the policy could be made more specific for certain time, and location attributes.

#### SDS Policy

Send sensing-data for <scan-task-L-band-User-Jim> to data-store <FooStore3>

PolicyID: <generated>

PolicyName: User-Jim-L-Band-DataStorePolicy

Policy-Category: SDS-Policy

PolicyDescription: Configure data store for a scan request.

Policy-Action: Transmit-sensing-data

SCOS-Task: scan-task-L-band-User-Jim

Resource: FooStore3

*Note: The data-store is specified with resource on which operation is done. The sensing-task is identified pre-defined using name-alias. Optionally, the policy could be made more specific for certain time, and location attributes.*

# Informative: Latency Requirements for Scans

The latency requirement for performing a sensing task and transmitting metadata from SD to DM is a critical metric for certain use cases.

The maximum allowed latency depends on signal type and frequencies that need to be scanned, and are determined by the application. As such, a number of reference applications are given, with recommended latencies. In each case, to meet the specified latency, the SCOS system design and particular implementation would need to be capable.

Maximum Latency would be the sum of:

* Task Request Latency: Time from scan request to scan start
* Scan Time Latency: Time from scan start to scan complete
* Spectrum Characterisation Latency: Time from scan complete to algorithmic processing
* Sensing Data Delivery Latency: Time from processing to delivery to Data Consumer

|  |  |  |
| --- | --- | --- |
| **Reference Application** | **Band Swept** | **Maximum Latency** |
| TVWS Base Station Device | 460-760MHz |  |
| CBRS Base Station Device | 3.5GHz-3.8GHz |  |
| TVWS CPE Device | 460-760MHz |  |
| CBRS CPE Device | 3.5GHz-3.8GHz |  |
| Etc | 460-760MHz |  |
|  |  |  |

# Informative: Regulatory Technical requirements

Various countries will have differing requirements here, but a few countries already have definitions in place that should be observed. For example, in the FCC rules for the VHF/UHF TV bands, the FCC requires a spectrum sensing detection accuracy as specified by the Table 1: FCC Sensing sensitivity requirements.

**Table 1: FCC Sensing sensitivity requirements**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Regulatory domain** | **Type of signal** | **Sensing detection threshold**  **(in dBm)** | **Data fusion rule for distributed sensinga** | **Monitoring requirements** |
| USA | ATSC | –114  (averaged over 6 MHz) | “OR” rule | Detection threshold referenced to an omni-directional receive antenna with a gain of 0 dBi |
| USA | NTSC | –114  (averaged over 100 kHz) | “OR” rule | Detection threshold referenced to an omni-directional receive antenna with a gain of 0 dBi |
| USA | Wireless microphone | –107  (averaged over 200 kHz) | “OR” rule | Detection threshold referenced to an omni-directional receive antenna with a gain of 0 dBi |

aThe value “1” indicates detection.

Other requirements for the 2.7 GHz to 3.7 GHz band shall be defined based on the evolving regulations. For example, the spectrum sensing devices in the 2.7 GHz to 3.7 GHz can sense for Radar Signals and provide that information to the Spectrum Access System (SAS) that is being defined in these bands.

# Device and System Security Recommendations

\* Remote access to SD hardware through remote secure shell (SSH) and similar technologies must not use the same keys as TM/SD interface keys

\* Devices’ physical characteristics must be evaluated and enumerated at build validation and testing, with hardware and configuration parameters written to file /DEVICEHARDWAREPARAMETERS.CONFIGFILE (placeholder) and stored in non-writable file

\* Any changes to hardware configuration (e.g. change of antenna) must be recorded in DEVICEHARDWARECONFIGCHANGES.LOGILE and changes made to relevant parameters in D..H...P...CONFIGFILE, either through manual editing of config file or through a secure remote update mechanism (e.g. scripted SCP file revision).

# Radio performance requirements

## Sensitivity and Noise

# (informative) Sensing

This annex contains descriptions of a number of sensing techniques. A sensing technique is an implementation of the spectrum sensing function.

# References

# (informative) Bibliography

Bibliographical references are resources that provide additional or helpful material but do not need to be understood or used to implement this standard. Reference to these resources is made for informational use only.

1. The Institute of Electrical and Electronics Engineers, Inc.

   3 Park Avenue, New York, NY 10016-5997, USA

   Copyright © 2017 by The Institute of Electrical and Electronics Engineers, Inc.

   All rights reserved. Published <Date Published>. 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 978-0-XXXX-XXXX-XSTDXXXXX

   Print: ISBN978-0-XXXX-XXXX-XSTDPDXXXXX

   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. [↑](#footnote-ref-1)
2. IEEE Standards Dictionary Online is available at: <http://dictionary.ieee.org> [↑](#footnote-ref-2)