INTERNATIONAL STANDARD

ISO/IEC/IEEE 18882

First edition
2017-02

Information technology — Telecommunications and information exchange between systems — Ubiquitous green community control network: Heterogeneous networks convergence and scalability

Technologies de l'information — Télécommunications et échange d'information entre systèmes — Protocole de contrôle de la communauté verte omniprésente: convergence et extensibilité de réseaux hétérogènes

Reference number
ISO/IEC/IEEE 18882:2017(E)

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

The main task of ISO/IEC JTC 1 is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is called to the possibility that implementation of this standard may require the use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. ISO/IEEE is not responsible for identifying essential patents or patent claims for which a license may be required, for conducting inquiries into the legal validity or scope of patents or patent claims or determining whether any licensing terms or conditions provided in connection with submission of a Letter of Assurance or a Patent Statement and Licensing Declaration Form, 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 ISO or the IEEE Standards Association.

ISO/IEC/IEEE 18882 was prepared by the Corporate Advisory Group of the IEEE SA Board of Governors (as IEEE Std 1888.2-2014). It was adopted by Joint Technical Committee ISO/IEC JTC 1, *Information technology,* Subcommittee SC 6, *Telecommunications and information exchange between systems*, in parallel with its approval by the ISO/IEC national bodies, under the “fast-track procedure” defined in the Partner Standards Development Organization cooperation agreement between ISO and IEEE. IEEE is responsible for the maintenance of this document with participation and input from ISO/IEC national bodies.

IEEE Standard for Ubiquitous Green Community Control Network: Heterogeneous Networks Convergence and Scalability

Sponsor

Corporate Advisory Group
of the
IEEE-SA Board of Governors

Approved 27 March 2014

IEEE-SA Standards Board

Abstract: This standard describes heterogeneous networks convergence and scalability, specifies the requirements of network convergence, extends the system architecture defined in IEEE Std 1888™, IEEE Standard for Ubiquitous Green Community Control Network Protocol, with two new IEEE 1888™ Components, i.e., the reconfigurable resolution server (RRS) and the intelligent application resolver (IAR), and generalizes primitive data type expressions and explicit field-bus data type management in IEEE 1888 systems. This standard enables IEEE 1888 systems to interoperate with heterogeneous access networks efficiently and improves the efficiency, flexibility, scalability and manageability of IEEE 1888 systems.

Keywords: field-bus data type, heterogeneous network convergence, IEEE 1888.2™, intelligent application resolver, reconfigurable resolution server, scalability, primitive data type

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

PDF: ISBN 978-0-7381-9026-6 STD98593
Print: ISBN 978-0-7381-9027-3 STDPD98593

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 Notice" or "Important Notices and Disclaimers Concerning IEEE Standards Documents."

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. Volunteers 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 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-SA Website at http://ieeexplore.ieee.org/xpl/standards.jsp 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 IEEE standard was completed, the IEEE 1888.2 Working Group had the following membership:

Hongke Zhang, *Chair*

Beijing Jiaotong University
BII Group Holdings Ltd.
China Telecommunications Corporation
Cisco Systems Inc.
Qingdao Gaoxiao Information Industry Co., Ltd.
Raisecom Technology Co., Ltd
The University of Tokyo

The P1888.2 Working Group gratefully acknowledges the contributions of the following participants. Without their assistance and dedication, this standard would not have been completed.

Changhe Du
Hiroshi Esaki
Ming Feng
Shuai Gao
Chen Gu
Xiaochuan Gu
Hongyuan Jiang
Lianshan Jiang
Wenjie Li
Zhigang Li
Dong Liu
Tsuyoshi Momose
Hideya Ochiai
Ming Qiu
Shoichi Sakane
Wei Su
Hongchao Wang
Hai Yang

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

Beijing Jiaotong University
Beijing University of Posts and Telecommunications (BUPT)
BII Group Holdings Ltd.
China Datang Corporation
China Telecommunications Corporation
Cisco Systems, Inc.
Guangzhou Smart Home Technology Standards Promotion Center
Guangzhou Video-star Electronics Industrial Co., Ltd.
Institute of Electrical Engineering Chinese Academy of Sciences
Institute of Electronic and Information Engineering in Dongguan (UESTC)
Institute of Software, Chinese Academy of Sciences (CAS)
Marvell Semiconductor, Inc.
Qingdao Gaoxiao Information Industry Co., Ltd.
Raisecom Technology Co., Ltd.
Renesas Electronics Corporation
Southwest Jiaotong University
State Grid Corporation of China (SGCC)
The University of Tokyo
Tsinghua University
ZTE Corporation

When the IEEE-SA Standards Board approved this standard on 27 March 2014, it had the following membership:

John Kulick, *Chair*
Jon Walter Rosdahl, *Vice Chair*
Richard H. Hulett, *Past Chair*
Konstantinos Karachalios, *Secretary*

Peter Balma
Farooq Bari
Ted Burse
Clint Chaplin
Stephen Dukes
Jean-Philippe Faure
Gary Hoffman
Michael Janezic
Jeffrey Katz
Joseph L. Koepfinger*
David J. Law
Hung Ling
Oleg Logvinov
Ted Olsen
Glenn Parsons
Ron Petersen
Adrian Stephens
Peter Sutherland
Yatin Trivedi
Phil Winston
Don Wright
Yu Yuan

*Member Emeritus

Also included are the following nonvoting IEEE-SA Standards Board liaisons:

Richard DeBlasio, *DOE Representative*
Michael Janezic, *NIST Representative*

Patrick Gibbons
IEEE Standards Program Manager, Document Development

Krista Gluchoski
IEEE Standards Project Specialist, Professional Services

Joan Woolery
IEEE Standards Program Manager, Technical Program Development

Introduction

This introduction is not part of IEEE Std 1888.2™-2014, IEEE Standard for Ubiquitous Green Community Control Network: Heterogeneous Networks Convergence and Scalability.

IEEE 1888™ has enabled access interoperability with many standard and proprietary protocols for field-bus systems including BACnet™[a], LonWorks®[b], Modbus-based systems, ZigBee®[c] devices, etc. However, IEEE 1888 lacks translation schemes among different application data types, generalized primitive data type expressions, and ID mapping configuration between field-bus and IEEE 1888 systems. That is, there are no efficient and scalable solutions for heterogeneous network convergence in IEEE 1888 systems.

This standard aims to provide the standard criteria for network convergence and scalability that enhances the heterogeneous networks interconnection and improves the efficiency, flexibility, scalability and manageability of IEEE 1888.

This standard extends the system architecture defined in IEEE Std 1888™ [d], IEEE Standard for Ubiquitous Green Community Control Network, with two new Components, i.e., the reconfigurable resolution server (RRS) and the intelligent application resolver (IAR). With the RRS, IEEE 1888 systems can support remote and dynamic distribution of ID mapping configuration and translation rules. The IAR can perform automatic translation among different application data types. In addition, this standard generalizes primitive data type expressions, explicit field-bus data type management, and ID mapping configuration between field-buses and IEEE 1888 systems for heterogeneous networks convergence.

This document is organized as follows:

— Clause 4 identifies the background and the requirements that this standard enables.
— Clause 5 defines the IEEE 1888 system architecture with the RRS and the IAR.
— Clause 6 generalizes the primitive data type expression.
— Clause 7 defines the management rule for importing field-bus data type.
— Clause 8 describes the security consideration.

[a] BACnet® is a registered trademark of American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE).
[b] LonWorks® is a registered trademark of Echelon Corporation.
[c] ZigBee® is a registered trademark of the ZigBee Alliance.
[d] Information on references can be found in Clause 2.

Contents

IEEE Standard for Ubiquitous Green Community Control Network: Heterogeneous Networks Convergence and Scalability

1. Overview

1.1 Scope

Based on the protocol defined in IEEE Std 1888™[1], IEEE Standard for Ubiquitous Green Community Control Network Protocol, this standard extends component and data type definitions, message formats, and communication procedures for heterogeneous network convergence and scalability. This standard also describes heterogeneous networks interconnection issues and requirements. Also, this standard specifies system architecture and solutions to improve heterogeneous networks convergence and scalability while offering system robustness and supplying better performance in system operation and management.

1.2 Purpose

This standard describes the standard criteria for network convergence and scalability that enhances the Ubiquitous Green Community Control Network (UGCCNet) heterogeneous networks interconnection. This standard provides enhanced efficiency, flexibility, and scalability to construct a secure, manageable, and compatible system.

[1] Information on references can be found in Clause 2.

2. Normative references

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

IEEE Std 1888™, IEEE Standard for Ubiquitous Green Community Control Network Protocol.[2, 3]

IEEE Std 1888.3™, IEEE Standard for Ubiquitous Green Community Control Network Protocol: Security.

XML Schema Part 2: Datatypes Second Edition, P.V. Biron and A. Malhotra, eds., October 2008.[4]

3. Definitions, abbreviations, and acronyms

3.1 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. [5]

application data type: A data type strongly associated to a specific application or a specific field-bus.

application data type domain: A set of application data types defined by a specific organization.

application domain: An alternative expression of application data type domain.

field-bus: An access network for sensors and actuators used in the field-levels. The term itself includes physical links, network-layer protocols, and application protocols on the physical links.

field-bus data type: An application data type associated to a specific field-bus.

ID mapping configuration: A content object that defines the binding between IEEE 1888™ Point IDs and field-bus level identifiers (e.g., identifiers for sensors and actuators on a field-bus).

translation rule: A content object that defines value projections from a set of application data types to another set of application data types between different application domains.

3.2 Abbreviations and acronyms

APP application

CSV comma-separated values

CUI character-user interface

---

[2] The IEEE standards or products referred to in this clause are trademarks of The Institute of Electrical and Electronics Engineers, Inc.
[3] IEEE publications are available from The Institute of Electrical and Electronics Engineers, 445 Hoes Lane, Piscataway, NJ 08854, USA (http://standards.ieee.org/).
[4] Available at http://www.w3.org/TR/xmlschema-2/.
[5] *IEEE Standards Dictionary Online* subscription is available at: http://www.ieee.org/portal/innovate/products/standard/standards_dictionary.html.

DNS	domain name system
GUI	graphical-user interface
GW	gateway
HVAC	heating, ventilating, air-conditioning
IAR	intelligent application resolver
NAT	network address translation
PLC	programmable logic controller
RRS	reconfigurable resolution server
UGCCNet	Ubiquitous Green Community Control Network
URI	uniform resource identifier
XML	eXtensible markup language

4. Background and requirements

4.1 Background issues

4.1.1 Ungeneralized ID mapping configuration between field-bus and IEEE 1888 systems at Gateway (GW)

Gateway (GW) products can have different configuration interfaces by vendors. They might provide a graphical-user interface (GUI) or a character-user interface (CUI) for a configuration interface. The format of configuration might be comma-separated values (CSV), eXtensible markup language (XML), or any other format. Currently, there are no common interfaces and no data format for configuring ID mapping between field-buses and IEEE 1888 systems, thus ID mapping cannot be managed generally and configured remotely.

4.1.2 Various application data types co-exist in IEEE 1888

IEEE 1888 has enabled access interoperability with many standard and proprietary protocols for field-bus systems including building automation and control networks (such as BACnet™[6] [B1][7]), local operating networks (such as LonWorks®[8] [B3]), Modbus-based systems [B4], ZigBee®[9] devices [B6], ECHONET™[10] [B2], programmable logic controllers (PLCs), or other comparable networks. However, the

[6] BACnet® is a registered trademark of American Society of Heating, Refrigerating and Air-Conditioning Engineers (ASHRAE).
[7] The numbers in brackets correspond to those of the bibliography in Annex F.
[8] LonWorks® is a registered trademark of Echelon Corporation.
[9] ZigBee® is a registered trademark of the ZigBee Alliance.
[10] ECHONET™ is a registered trademark of the ECHONET Consortium.

definition of the contents and semantics of the data exchanged in IEEE Std 1888™ are still incomplete. For example, BACnet data and LonWorks data sometimes co-exist in the same IEEE 1888 data storage, but data representation is “1” for heating mode in BACnet and “HVAC_HEAT” in LonWorks. Thus, applications have to recognize such differences in order to correctly understand the meaning.

4.1.3 Value can have different expressions at a primitive level

Boolean, integer, real, and string are basic data types that appear in many field-bus systems. As IEEE Std 1888™ does not define even primitive data types, these data types might produce values in an unexpected expressions, or applications (APPs) might fail to parse the values if they assume only a small set of expressions.

4.1.4 Field-bus data types are not explicitly managed in IEEE 1888

Point ID in IEEE 1888 often represents sensors or actuators at the field-bus. Information about the types of data produced by such sensors is not explicitly described and managed in IEEE Std 1888. Thus, system integrators need to manage such data types when developing and re-configuring an IEEE 1888 system.

4.2 Requirements and design principles

4.2.1 Allow remote configuration of the IEEE 1888 GW for binding field-buses and IEEE 1888 systems

This standard defines the interface and format of configuring IEEE 1888 GWs regarding ID mapping between field-bus and IEEE 1888 systems, i.e., mapping between field-bus IDs and Point IDs. This enables remote configuration of the GW with the change of the field-bus level configuration—adding new sensors at the field-bus. This standard recommends CSV formatting and provides a sample in Annex A.

4.2.2 Allow value translation among different application data types

Some applications are specially designed for specific application data types (e.g., Vendor A’s BACnet data types). It is not always easy for such concretely designed applications to adapt to heterogeneous data types (e.g., not only BACnet data types, but also for LonWorks data types, ECHONET data types, and other proprietary data types). There are some common primitives (e.g., HVAC heating mode) in meta definition, which means that different representatives are interchangeable with each other. Of course, some application data types cannot be cast to the other data type systems. In this standard, it should allow mapping of a data type (e.g., from BACnet to LonWorks), and casting of data to the other data types. For example, it should allow the translation of “1” (BACnet representation) into “HVAC_COOL” (LonWorks representation).

4.2.3 Allow remote configuration of translation rule

This standard should define the interface and format of translation rule to allow remote configuration of translation. As the recommended format for translation rules, this standard provides a CSV formatting case in Annex B.

4.2.4 Define primitive data types commonly existing in heterogeneous networks

There are some common primitive data types. For example, Boolean type, integer type and real type are commonly used in many heterogeneous networks. This standard defines those primitive data types and their expressions, which should be adopted commonly in IEEE 1888 systems. This standard focuses on a set of primitive data types because it is not feasible to define comprehensive data types that cover all heterogeneous field-bus protocols. This definition allows us to stand on the same primitive data format. The designers of GWs (for heterogeneous networks) should adopt this rule so as to increase interoperability at the primitive level.

4.2.5 Make field-bus data type explicit at Registry

A field-bus usually defines its own application data types. IEEE 1888 has already allowed the inclusion of those data transparently by GW architecture defined in IEEE Std 1888. However the standard itself does not define how to recognize those data types. This standard allows handling of those data and their data types. The approach is to define namespaces and attributes for those heterogeneous (and application-specific) data types. This standard provides presentation rules for some major field-bus data models in Annex C.

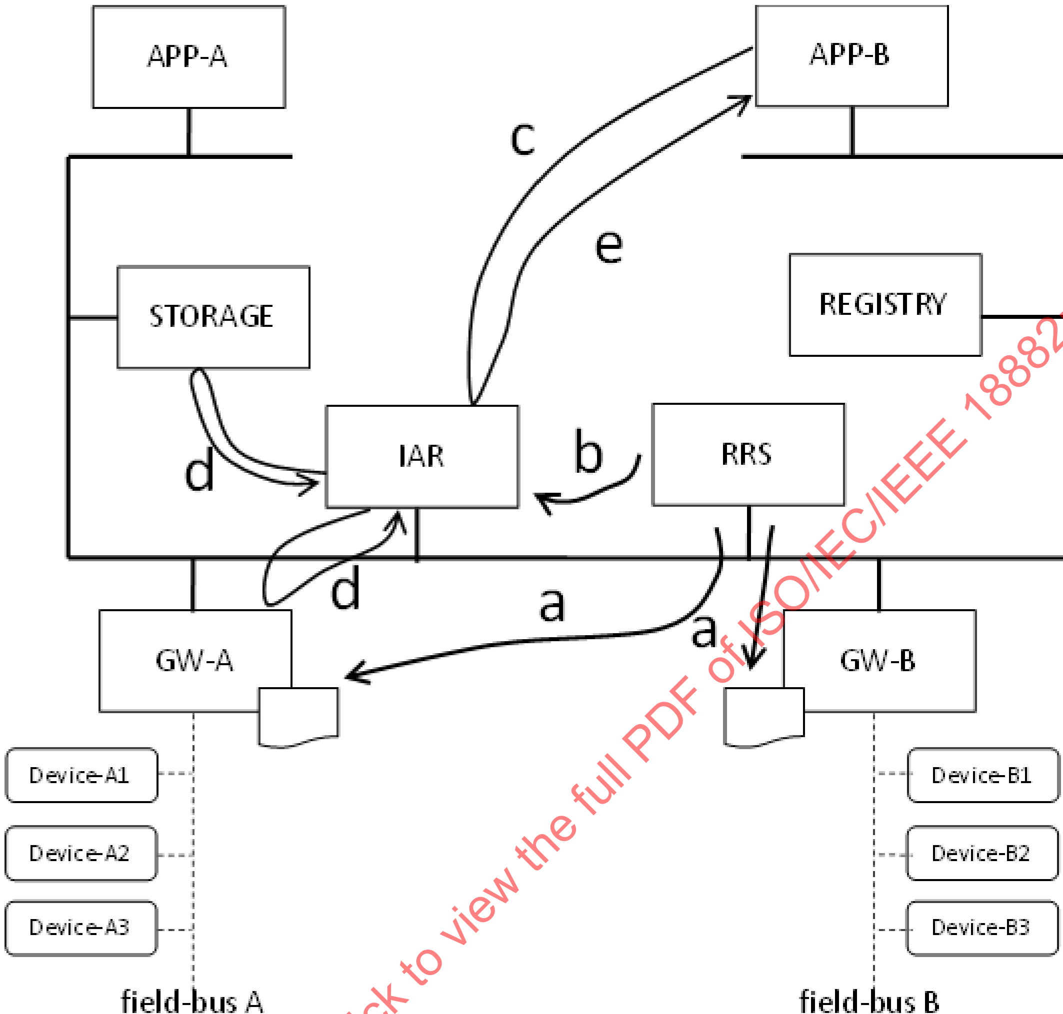
5. Architecture

5.1 Overview

To implement the requirements identified in the previous section, this standard introduces two new IEEE 1888 Components: the reconfigurable resolution server (RRS) and the intelligent application resolver (IAR). The definition of RRS and IAR are as follows:

- The RRS manages ID mapping configuration between field-bus and IEEE 1888 Point ID and value translation rules among different application data type domains.
- The IAR carries out value translation among different application data type domains.

Figure 1 shows how the RRS and the IAR typically work in the IEEE 1888 architecture. Note that the following description just provides an example; for a precise definition, see 5.2 and 5.3. The RRS manages the configuration of the GW (only ID mapping between field-bus and point ID) and the translation rule.



a) The RRS distributes ID mapping configuration to GWs

b) The RRS distributes the translation rule to the IAR.

The IAR carries out translation among different application data types. Here, the IAR does not necessarily need to be dynamically configured by the RRS. Manual configuration is also allowed. In this figure, the APP-B is an APP of domain B, the GW-A is a GW of domain A, and the Storage has the history record of domain A's data. APP-B can only understand the data defined in domain B. Here,

c) APP-B is asking the IAR to FETCH point IDs defined for domain B, which IDs are actually associated with point IDs defined for domain A.

d) As the IAR recognizes this association, it FETCHs values from GW-A (or Storage) with point IDs defined for domain A.

After translating the values FETCHed from GW-A (or STORAGE) into domain B's form,

e) The IAR returns to APP-B as the reply of c).

Figure 1—IEEE 1888 system architecture with RRS and IAR

Though in this figure IAR receives requests only from APP, the IAR can generally receive it from any other Components (e.g., GW, Storage). Although in this figure the IAR is independently depicted, it can be unified into GW, Storage, and any other Components (see 5.3) in practice.

Figure 1 only shows component-to-component communications. As for Registry-enabled operation, refer to 5.2.5 and 5.3.5.

5.2 Reconfigurable resolution server (RSS)

5.2.1 RSS overview

The role of the RRS is to distribute ID mapping configuration to the GW and translation rules to the IAR. This standard defines two procedures for this distribution: FETCH-based configuration and WRITE-based configuration. FETCH-based configuration is mandatory, but WRITE-based configuration is optional (see 5.2.3 and 5.2.4 for details). In the operation of this distribution, this standard allows co-existence of multiple RRSs in the network. This standard also defines dynamic binding between the RRS and the GW/IAR using Registry (see 5.2.5). As for the recommended format of ID mapping configuration and translation rules, see Annex A and Annex B.

5.2.2 Definition of Point IDs for mapping configuration and translation rules

ID mapping configuration and translation rules shall have the following Point IDs.

— ID Mapping Configuration: ID_THAT_REPRESENTS_GW / idMap

— Translation Rule: ID_THAT_REPRESENTS_IAR / translationRule

Here, ID_THAT_REPRESENTS_GW is the prefix of the Point ID, which represents the GW itself. For instance, if the configuration of GW A, B, and C are managed at an RRS, ID_THAT_REPRESENTS_GW can be defined as in Table 1:

Table 1—Definition example of ID_THAT_REPRESENTS_GW

Gateway	ID_THAT_REPRESENTS_GW
GW A	http://example.org/gwA/
GW B	http://example.org/gwB/
GW C	http://example.org/gwC/

In the same way, the operator can define ID_THAT_REPRESENTS_IAR as in Table 2:

Table 2—Definition example of ID_THAT_REPRESENTS_IAR

Gateway	ID_THAT_REPRESENTS_IAR
IAR A	http://example.org/iarA/
IAR B	http://example.org/iarB/
IAR C	http://example.org/iarC/

Here, if GW A and IAR A are unified (see 5.3), their ID_THAT_REPRESENTS_GW and ID_THAT_REPRESENTS_IAR can be the same. For example, if ID_THAT_REPRESENTS_GW is “http://example.org/gwA/”, ID_THAT_REPRESENTS_IAR can be “http://example.org/gwA/”.

In this case, the Point ID for ID mapping configuration and the translation rule for GW A (=IAR A) becomes as follows.

— ID mapping configuration: http://example.org/gwA/idMap

— Translation rule: http://example.org/gwA/translationRule

When distributing these configurations and rules, the RRS shall specify the “time” attribute in the “value” element in order to show the version of the configuration and rule information by time. For example,

```
<transport>
 <body>
  <point id=”http://example.org/gwA/idMap”>
   <value time=”2013-01-01T00:00:00+08:00”>
     Content of ID map – See Annex A.
   </value>
 </point>
 </body>
</transport>
```

At the GW or the IAR, if the time attribute has changed from the previously received time, it shall use the content as the configuration. Because if the GW or the IAR can only use the latest configuration (in other words, if they only accept updated configurations), and when the RRS has mistakenly sent a vastly postdated configuration, the RRS with the correct clock becomes unable to change the configuration of the GW or the IAR.

5.2.3 FETCH-based configuration

In FETCH-based configuration, the GW initiates a FETCH request to get ID mapping from the RRS. In the same manner, the IAR (if dynamic update of translation rule is enabled) can also initiate a FETCH request to get a translation rule from the RRS.

This procedure typically occurs at boot up. After that, they can periodically do this [especially deployed behind a network address translation (NAT)] in order to check the change of the configurations. In this procedure, FETCH requests shall specify attrName=”time” and select=”maximum” in the query key.

5.2.4 WRITE-based configuration

In a WRITE-based configuration, the RRS initiates a WRITE request to set the ID mapping configuration into the GW. In the same manner, the RRS can also initiate a WRITE request to set a translation rule into the IAR (if dynamic update of the translation rule is enabled).

The RRS executes this procedure typically when the administrator of the IEEE 1888 system changes one or part of one of the configurations. However, some GWs do not implement a WRITE server function or are deployed behind an NAT or firewall. Such GWs shall implement FETCH-based configuration.

5.2.5 Automatic binding between the RRS and the GW/IAR using Registry

5.2.5.1 Automatic binding overview

To make the distribution of ID mapping configuration and translation rules scalable, this standard allows the co-existence of multiple RRSs in the network. In this case, a manual configuration specifying remote IEEE 1888 components (i.e., RRS, GW, and IAR) makes engineering cost high. Instead, the system integrator should introduce Registry to automatically bind RRSs and GWs/IARs appropriately. In order to do this, registration and lookup RRS and GW/IAR should be performed as follows in 5.2.5.2 and 5.2.5.3.

5.2.5.2 Registration of RSS and GW/IAR

If the registry operation mode is enabled, the RRS shall register itself with the point IDs it manages with stream="in" specified. In this operation mode, the GW also registers itself with the Point ID= "ID_THAT_REPRESENTS_GW/idMap" with stream="out" specified, only if the GW implements the WRITE target function. The IAR also registers itself with the Point ID="ID_THAT_REPRESENTS_IAR/translationRule" with stream="out" specified.

For example, in the case of Table 1 and Table 2, the RRS registers itself by the following request message to the Registry.

```
<transport>
  <body>
   <component uri="http://rrs1.example.org/axis2/services/RRS" support="FETCH"  … >
     <key id="http://example.org/gwA/idMap" attrName="time" stream="in" />
     <key id="http://example.org/gwB/idMap" attrName="time" stream="in" />
     <key id="http://example.org/gwC/idMap" attrName="time" stream="in" />
     <key id="http://example.org/iarA/translationRule" attrName="time" stream="in" />
     <key id="http://example.org/iarB/translationRule" attrName="time" stream="in" />
     <key id="http://example.org/iarC/translationRule" attrName="time" stream="in" />
   </component>
  </body>
</transport>
```

This means that it manages the Point IDs above and produces the contents of them.

As for the GW, It registers itself by the following request message to the Registry.

```
<transport>
  <body>
   <component uri="http://gwa.example.org/axis2/services/GW" support="WRITE"  … >
     <key id="http://example.org/gwA/idMap" attrName="time" stream="out" />
   </component>
  </body>
</transport>
```

As for the IAR, it registers itself by the following request message to the Registry.

```
<transport>
  <body>
   <component uri=”http://iara.example.org/axis2/services/IAR” support=”WRITE”  … >
     <key id=”http://example.org/iarA/translationRule” attrName=”time” stream=”out” />
   </component>
  </body>
</transport>
```

If the GW and the IAR are unified, the IAR can register itself by the following request message to the Registry (in this case the RRS manages and registers both point IDs presented here).

```
<transport>
  <body>
   <component uri=”http://gwa.example.org/axis2/services/GW” support=”WRITE”  … >
     <key id=”http://example.org/gwA/idMap” attrName=”time” stream=”out” />
     <key id=”http://example.org/gwA/translationRule” attrName=”time” stream=”out” />
   </component>
  </body>
</transport>
```

5.2.5.3 Lookup of RSS and GW

If the registry operation mode is enabled, the RRS may lookup the GW from Registry with stream=”out” specified to set new ID mapping configuration to the corresponding GW by WRITE procedure. In the same manner, the RRS may lookup the IAR from Registry with stream=”out” specified to set a new translation rule to the corresponding IAR by WRITE procedure.

If the registry operation mode is enabled, the GW may also lookup appropriate RRS(s) with Point ID=”ID_THAT_REPRESENTS_GW/idMap” and stream=”in” specified in the key element of LOOKUP request, before getting the current ID mapping configuration from the resolved RRS(s). In the same manner, the IAR (if dynamic update of translation rule is enabled) may also lookup appropriate RRS(s) with Point ID=”ID_THAT_REPRESENTS_IAR/translationRule” and stream=”in” specified in the key element of LOOKUP request, before getting the current translation rule from the resolved RRS(s).

For example, in the case of Table 1, the RRS looks up the GW by the following request message to the Registry:

```
<transport>
  <header>
    <lookup id="…" type="component">
     <key id="http://example.org/gwA/idMap" attrName="time" stream="out" />
    </lookup>
  </header>
</transport>
```

Then, the Registry will reply as follows:

```
<transport>
  <header>
    <lookup id="…" type="component">
     <key id="http://example.org/gwA/idMap" attrName="time" stream="out" />
  </lookup>
  </header>
  <body>
   <component uri="http://gwa.example.org/axis2/services/GW" support="WRITE" … >
     <key id="http://example.org/gwA/idMap" attrName="time" stream="out" />
   </component>
  </body>
</transport>
```

In this way, the RRS can find how to access the GW (access information is presented at component "uri" attribute) and set ID mapping configuration by WRITE procedure. Figure 2 shows the entire procedure.

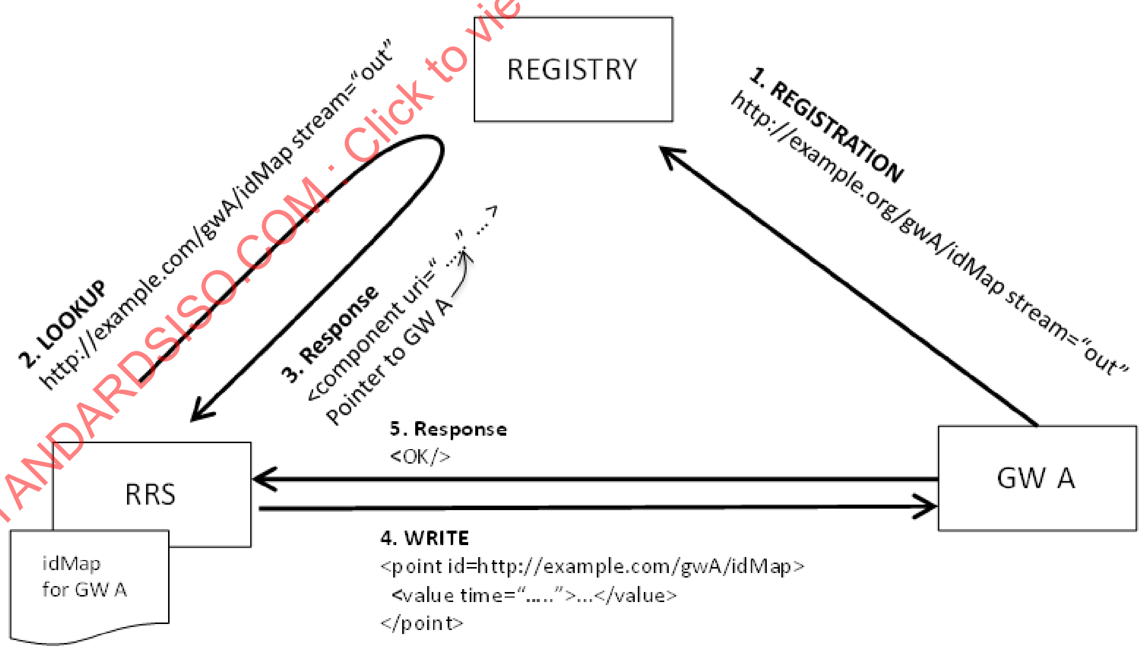


Figure 2—WRITE-based configuration using Registry

Here, the RRS can make lookup requests with multiple point IDs at the same time. In this case, Registry will reply with multiple Components (i.e., GW, IAR or a unified one). The RRS shall understand the response of Registry and make a WRITE request to those Components respectively.

As for the case when the GW initiates a LOOKUP request, the GW looks up the RRS by the following request message to the Registry.

```
<transport>
  <header>
    <lookup id="…" type="component">
     <key id="http://example.org/gwA/idMap" attrName="time" stream="in" />
    </lookup>
  </header>
</transport>
```

NOTE—Here, if the GW is unified with the IAR, it can specify the PointID for the translation rule at the key element in parallel.

Then, the Registry will reply as follows:

```
<transport>
  <header>
    <lookup id="…" type="component">
     <key id="http://example.org/gwA/idMap" attrName="time" stream="in" />
    </lookup>
  </header>
  <body>
   <component uri="http://rrs1.example.org/axis2/services/RRS" support="FETCH"  … >
     <key id="http://example.org/gwA/idMap" attrName="time" stream="in" />
   </component>
  </body>
</transport>
```

In this way, the GW can find an appropriate RRS and access information by component "uri" attribute and get ID mapping configuration from the RRS. Figure 3 shows the entire procedure.

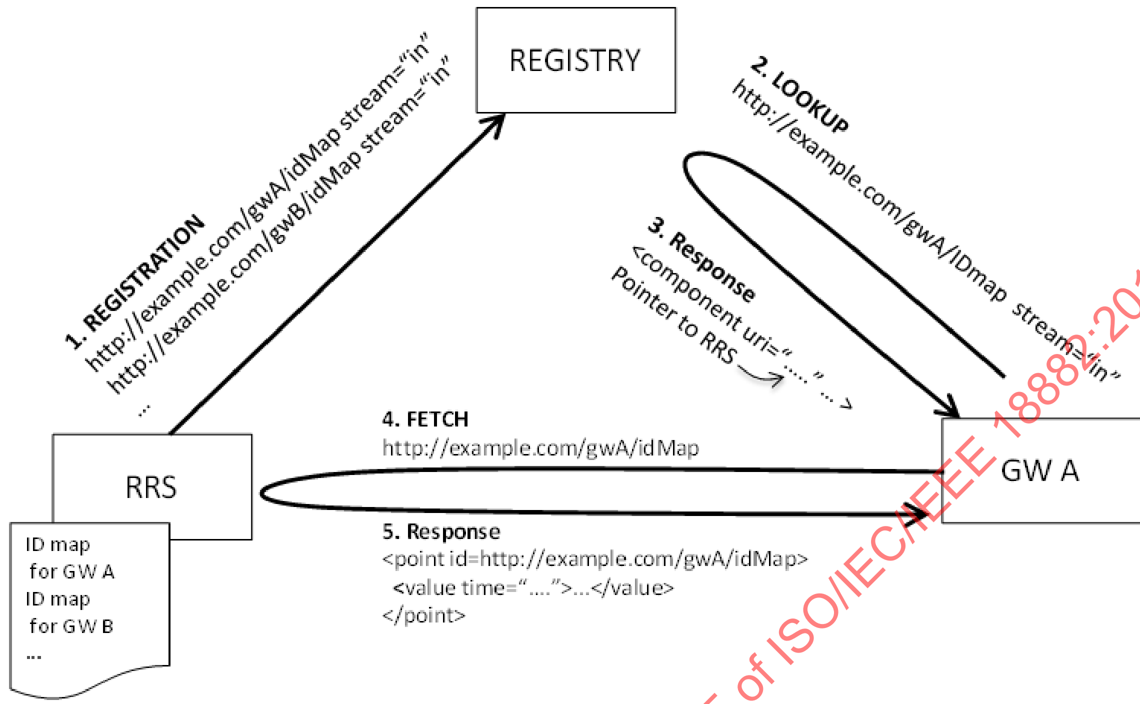


Figure 3—FETCH-based configuration using Registry

This procedure is also true to the IAR's case and the GW can also make such lookup requests with multiple point IDs at the same time, especially when the GW is unified with the IAR.

5.3 Intelligent application resolver (IAR)

5.3.1 IAR Overview

The role of the IAR is the translation of application data types from one application domain to another application domain. For example (see Figure 4), in Application A (at Field-Bus A's world), they use "1" to indicate HVAC cooling mode, but in Application B (at Field-Bus B's world), they use "COOL" to indicate HVAC cooling mode. Though they represent the same meaning, they use different data expressions. The IAR, working as a proxy, translates those different expressions between A and B: i.e., between 1 and "COOL".

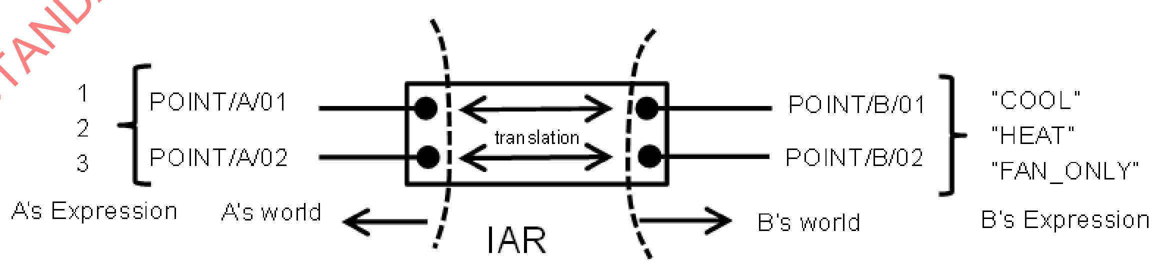


Figure 4—Model of an IAR

In Figure 4, there are bindings between POINT/A/01 and POINT/B/01 and between POINT/A/02 and POINT/B/02. (Here, the Point ID naming rule follows IEEE 1888). In this situation, when an APP FETCHs data of POINT/A/01 at the IAR, the IAR retrieves data of POINT/B/01 from B’s world, translates the expression, and returns to the APP as POINT/A/01. If the value of POINT/B/01 is “HEAT”, the APP gets “2” as the value.

The IAR should be configured with necessary information for this translation. This includes value mapping definition, arithmetic calculation formula, and Point ID-to-Point ID binding. The IAR also needs to know (if it does not use Registry) access information of their related GW, Storage, and APP. Annex B provides a sample configuration for value mapping, arithmetic calculation, and Point ID-to-Point ID binding.

Though this standard defines the IAR, typically it can be implemented (unified) with other types of Components (e.g., GW, Storage, and APP), as illustrated in Figure 5.

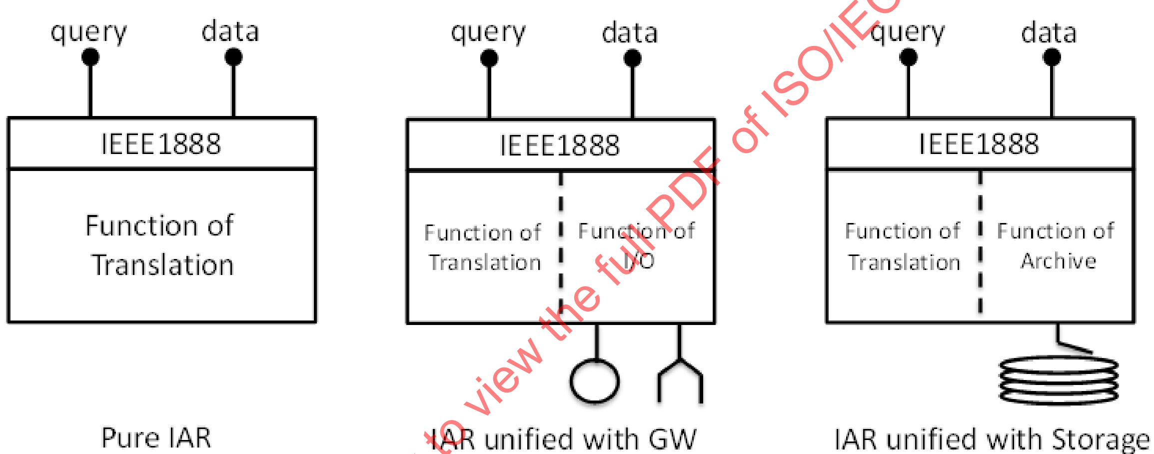


Figure 5—IAR-unified Components; various implementations of the IAR

5.3.2 The IAR as a FETCH provider

Figure 6 illustrates how the IAR works as a FETCH provider after it receives a FETCH request from a FETCH requester.

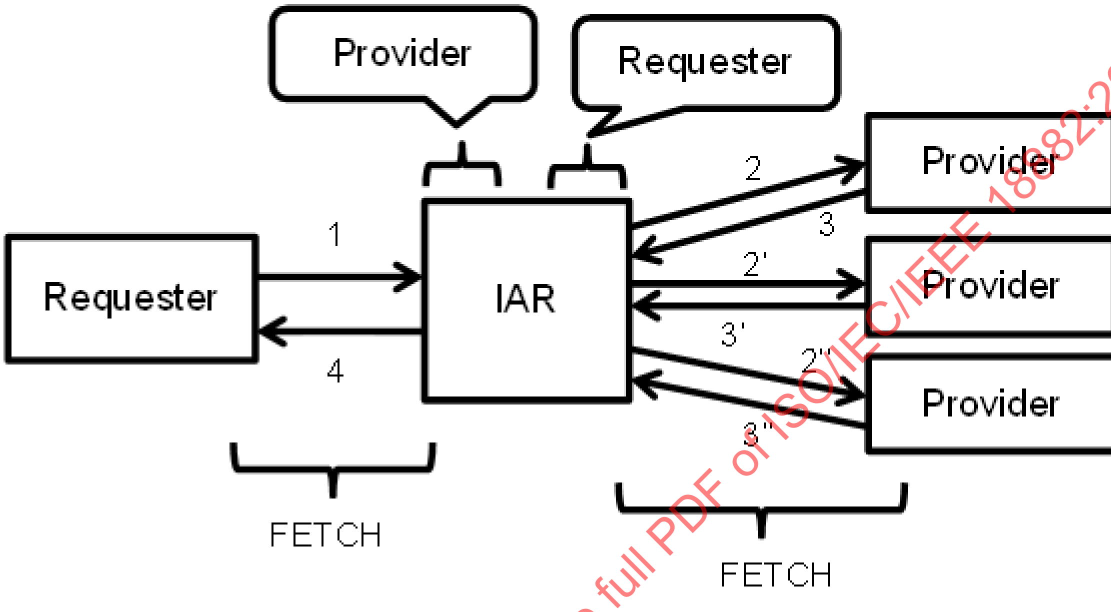


Figure 6—The IAR as a FETCH provider

When (1) the IAR receives a FETCH request (invoked query method with type=”storage”), it looks up the corresponding Point IDs and access uniform resource identifiers (URIs) of the Components that have the data requested. Here, the configuration information should be locally managed or the access URIs of Components could be resolved using Registry (if the Registration operation is enabled). (2)(2’)(2’’) Then, the IAR carries out FETCH requests to those Components (FETCH providers), and (3)(3’)(3’’) receives the contents of data. After translating the data, (4) the IAR replies to the FETCH requester as the response of the FETCH request.

The IAR will reply with an error with type=” TRANSLATION_FAILED” if it cannot perform the requested translation.

The IAR will reply with an error with type=”NESTED” if it receives an error from the Target. The error received should be presented in the content of the error element.

The IAR should reply “cursor” attributes with the contents of translated data if there remain more data to be retrieved, as the specification of IEEE 1888 defines. Here, IAR can just relay the “acceptableSize” attributes from the FETCH requester to the FETCH providers. This will help the implementation of the IAR. If the IAR does not implement the “cursor” function but still has more data to reply that cannot be accepted by FETCH requester, the IAR shall reply with an error with type=”TOO_MANY_VALUES”.

5.3.3 The IAR as a WRITE target

Figure 7 illustrates how the IAR acts as a WRITE target after it receives a WRITE request from a WRITE requester.

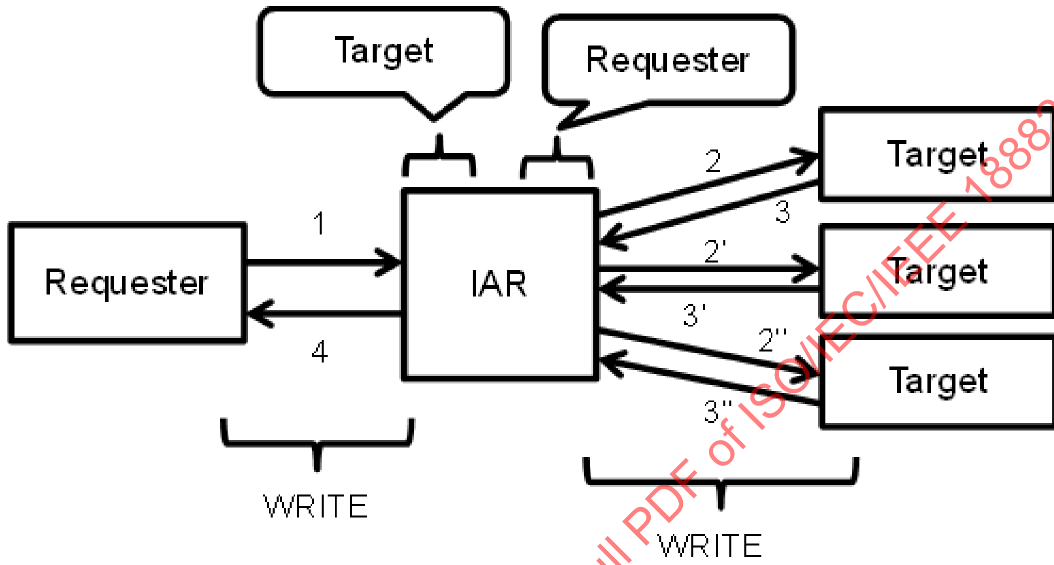


Figure 7—The IAR as a WRITE target

When (1) the IAR receives a WRITE request (invoked data method), it looks up the corresponding Point IDs and access URIs of the Components that in charge of the data posted. The configuration information should be locally managed, or the access URIs of the Components could be resolved using Registry (if the Registration operation is enabled). Then, (2)(2')(2'') the IAR carries out WRITE requests to those Components (WRITE targets) with translated data and checks (3)(3')(3'') the result (OK or NG) from the targets. If the IAR receives OK from all the targets, (4) it replies OK to the WRITE requester as the response.

The IAR will reply with an error with type="TRANSLATION_FAILED" if it cannot perform the requested translation.

The IAR will reply with an error with type="NESTED" if it receives an error from the Target. The error received should be presented in the content of the error element.

5.3.4 The IAR as a TRAP provider

5.3.4.1 IAR as TRAP provider overview

Figure 8 illustrates how the IAR works as a TRAP provider after it receives a TRAP request from a TRAP requester.

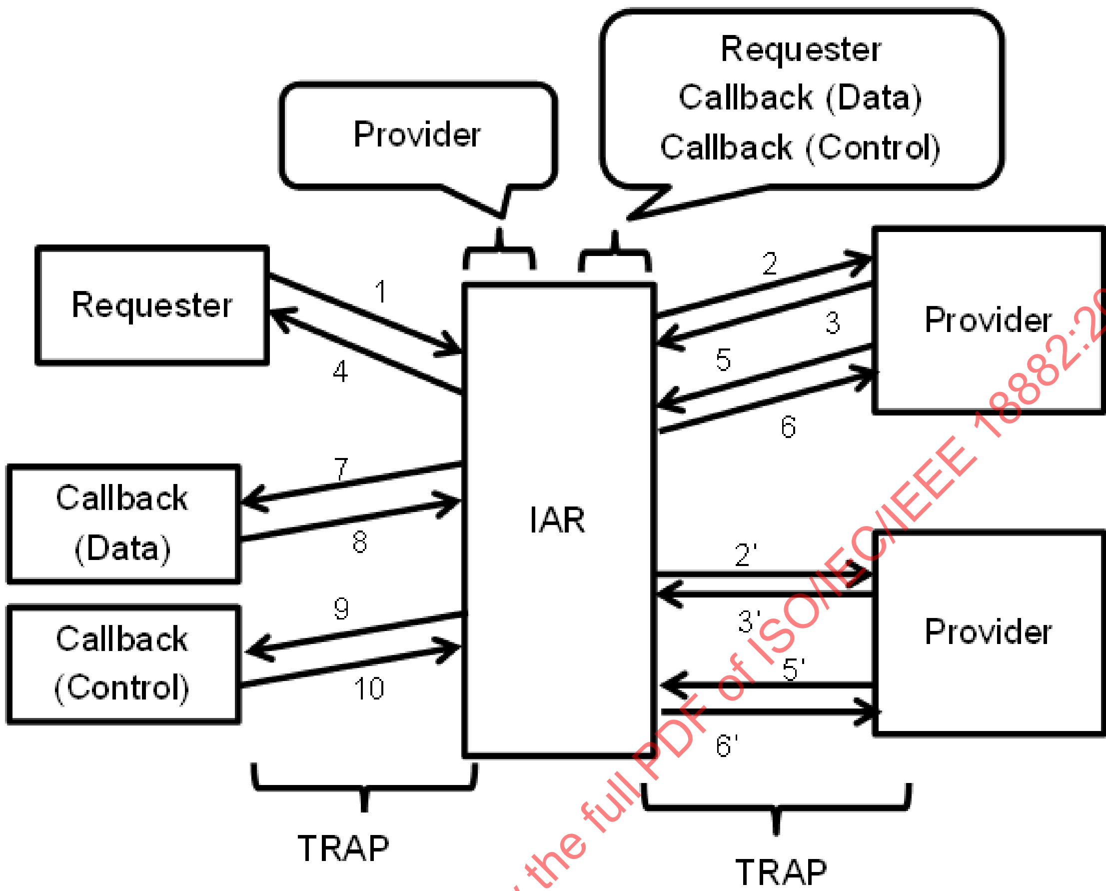


Figure 8—The IAR as a TRAP provider

5.3.4.2 TRAP request phase (1 – 4 in Figure 8)

When (1) the IAR receives a TRAP request (invoked data method), it looks up the corresponding Point IDs and access URIs of the Components that have the data requested. The configuration information should be locally managed, or the access URIs of the Components could be resolved using Registry (if the Registration operation is enabled). Then, (2)(2') the IAR carries out TRAP requests to those Components (TRAP providers) with the callbackData and callbackControl attributes to point itself in the network, and checks (3)(3') the result (OK or NG). If the IAR receives OK from all the providers, (4) it replies OK to the TRAP requester as the response.

The IAR will reply with an error with type="TRANSLATION_FAILED" if it cannot perform the requested translation.

IAR will reply with an error with type="NESTED" if it receives an error from the Target. The error received should be presented in the content of the error element.

This TRAP request should be made repeatedly (because the TTL of the TRAP request expires at the provider) as IEEE 1888 defines.

5.3.4.3 TRAP callback phase (5 – 10 in Figure 8)

If TRAP is requested, (5)(5') TRAP providers shall callback data to the IAR if they have updates that satisfy the TRAP request. Then, (6)(6') the IAR replies OK to the providers that made such callback. The IAR translates the data and map to the corresponding Point IDs, and (7) callback this update to the Callback (Data).

The TRAP providers (5)(5') can also callback control signals (typically, error information) to the IAR if they have an accident that should be notified. Then, (6)(6') the IAR replies OK to the TRAP providers. If the control signal is OK, (9) the IAR just sends OK to the Callback (Control). If the control signal is an error, then (9) the IAR sends an error with type="NESTED" to the Callback (Control). Here, the error received should be presented in the content of the error element. The Callback (Control) (10) replies OK to the IAR.

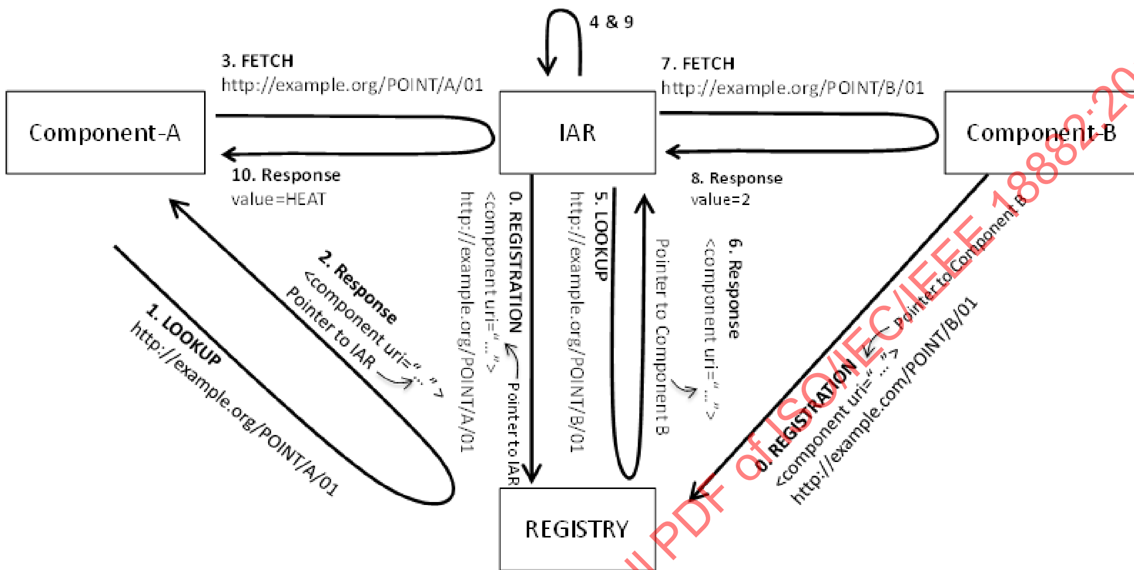
5.3.5 Automatic binding between Components and IAR using Registry

5.3.5.1 Registry automatic binding between Components and IAR overview

Figure 9 shows the procedure for automatic binding between Components and the IAR using Registry. If the Registration operation is enabled, IEEE 1888 Components register themselves to their Registry. The IAR shall also register itself to the Registry in this operation mode. Component-A, a requester, initiates a Component-to-Component communication (3. FETCH in Figure 9) after looking up the IAR at the Registry (1. LOOKUP and 2. Response). Then, the IAR initiates another Component-to-Component communication (7. FETCH and 8. Response) after looking up the target Components at the Registry (5. LOOKUP and 6. Response), referring to the value map loaded in their configuration (4. Value Mapping). The other procedures are the same as 5.3.2 (IAR as a FETCH requester), 5.3.3 (IAR as a WRITE requester), and 5.3.4 (IAR as a TRAP requester) define.

Value Mapping between Point IDs

http://example.org/POINT/A/01	http://example.org/POINT/B/01
COOL	1
HEAT	2
AUTO	3



NOTE 1—It is assumed that Component-A knows http://example.org/POINT/A/01. There would be several implementations for providing the point ID. In typical operation, it should be pre-configured locally. If the point has application-specific attributes and those attributes are registered at the Registry, the Component could find the point by making a point lookup request with the application-specific attributes dynamically. In addition, any other methods (using any databases or Web technologies) could be possible.

NOTE 2—Component-B can contain both http://example.org/POINT/A/01 and http://example.org/POINT/B/01 if it has translation capability (i.e., IAR function). When Component-A looks up http://example.org/POINT/A/01, in this case, the registry shall reply the access URL of Component-B, and Component-A directly accesses Component-B.

NOTE 3—If there are multiple application data domains, for example A, B and C (let's assume Component-C manages C's data), the binding among them (whether the binding is between A and B or between A and C) would be made by system integrators at the deployment phase. System integrators should typically develop the configuration of binding like this:

POINT/A/01/01 <-> POINT/B/01
POINT/A/01/02 <-> POINT/B/02
POINT/A/01/02 <-> POINT/B/02
POINT/A/02/01 <-> POINT/C/01
POINT/A/02/02 <-> POINT/C/02
POINT/A/02/02 <-> POINT/C/02

POINT/A/01 itself cannot represent both POINT/B/01 and POINT/C/01 at the same time because they are associated with different objects. Thus, system integrators allocate "POINT/A/01/01 for POINT/B/01" and "POINT/A/02/01 for POINT/C/01." Then this mapping table is installed into the IAR, and Registry knows that "POINT/B/*" is handled by Component-B and that "POINT/C/*" is handled by Component-C. Then, the IAR can perform the translation based on the configuration above.

Figure 9—Automatic binding between Components and IAR using Registry

NOTE 4—This architecture allows scalable operation of the IAR: i.e., a) the processing of, for example, 10 million different requests from different IEEE 1888 Components and each of them is asking different point IDs, and b) the processing of, for example, 10 million same requests (i.e., targeting at the same Point ID) from different IEEE 1888 Components. For the first case, the system can achieve scalability by delegating a subset of translation rules to multiple IARs in a distributed manner. Registry can manage which IAR is capable of translation for a certain translation requests, and, consequently, it distributes the load. For the latter case, the system can achieve scalability by deploying the same multiple IARs on the network and distribute the requests. By deploying multiple IARs with the same value mapping configurations, the system can multiply the capacity of processing the number of requests per second. Requesters (i.e., Component-A) choose one of them by letting the Registry manage such multiple IARs on the network. Here, requesters shall access the same server (i.e., the same IP address) if they use “cursor” or make “TRAP” requests for communicating with an IAR of them. In this way, the loads of Requests shall be balanced on them, thus it achieves scalability.

Figure 9—Automatic binding between Components and IAR using Registry *(continued)*

5.3.5.2 Registration of the IAR

In the example above, the registration request message to the Registry would be:

```
<transport>
 <body>
<component uri=”http://example.org/axis2/services/IAR” support=”…” …>
 <key id=”http://example.org/POINT/A/01” attrName=”time” RANGE_SPECIFICATION />
</component>
</body>
</transport>
```

RANGE_SPECIFICATION depends on the time-range of the original data or implementation of the IAR. It should be configured at the IAR, or the IAR can lookup the Registry (with original Point ID, i.e., http://example.org/POINT/B/01 in this case) to find available time range.

NOTE—If the point ID(s) has application-specific attributes, it could be registered at the Registry (not necessarily by the IAR, but by any Components that know the point ID).

5.3.5.3 Lookup of the IAR

In the example above, the lookup request message to the Registry would be:

```
<transport>
  <header>
 <lookup id=”…” type=”component”>
<key id=”http://example.org/POINT/A/01” attrName=”time” stream=”in”/>
 </lookup>
  </header>
</transport>
```

Then, the Registry replies as follows:

```
<transport>
  <header>
 <lookup id=”…” type=”component”>
<key id=”http://example.org/POINT/A/01” attrName=”time” stream=”in”/>
 </lookup>
  </header>
<body>
<component uri=”http://example.org/axis2/services/IAR” support=”…” …>
 <key id=”http://example.org/POINT/A/01” attrName=”time” RANGE_SPECIFICATION />
</component>
</body>
</transport>
```

RANGE_SPECIFICATION depends on the time-range of the original data or the implementation of IAR.

NOTE—If the point ID has application-specific attributes, the Component could find the point ID by making a point lookup request to the Registry with the application-specific attributes before making Component(=IAR) lookup request shown above.

6. Primitive data type

6.1 Primitive data type definition

In order to generalize the primitive data type expression, this standard strongly recommends using XML Schema Part 2: Datatypes Second Edition, which defines the data types boolean, int, unsignedInt, decimal, float, string, base64Binary, dateTime, duration, … and their canonical forms. For example, the following expressions to mean boolean “true” would be used:

```
<value time=”2013-04-29T00:00:00+08:00”>true</value>
<value time=”2013-04-29T00:00:00+08:00”>True</value>
<value time=”2013-04-29T00:00:00+08:00”>TRUE</value>
<value time=”2013-04-29T00:00:00+08:00”>T</value>
<value time=”2013-04-29T00:00:00+08:00”>t</value>
<value time=”2013-04-29T00:00:00+08:00”>1</value>
```

6.2 Primitive data type expressions in IEEE 1888

6.2.1 Boolean type

This standard should use Boolean type as `http://www.w3.org/2001/XMLSchema#boolean` defines.

For example, the expressions in the content of value element are:

```
<value time=”2013-05-01T00:00:00+08:00”>true</value>
<value time=”2013-05-01T00:01:00+08:00”>false</value>
```

6.2.2 Int type

This standard should use Int type as `http://www.w3.org/2001/XMLSchema#int` defines.

For example, the expressions in the content of value element are

<value time=”2013-05-01T00:00:00+08:00”>826</value>
<value time=”2013-05-01T00:01:00+08:00”>2147483647</value>
<value time=”2013-05-01T00:02:00+08:00”>0</value>
<value time=”2013-05-01T00:03:00+08:00”>-124</value>
<value time=”2013-05-01T00:04:00+08:00”>-2147483648</value>

The maximum value of int type is 2147483647, and the minimum is -2147483648. If it exceeds the boundary, long or integer type should be considered for use (see 6.2.10).

6.2.3 UnsigedInt type

This standard should use UnsignedInt type as `http://www.w3.org/2001/XMLSchema#unsignedInt` defines.

For example, the expressions in the content of value element are:

<value time=”2013-05-01T00:00:00+08:00”>826</value>
<value time=”2013-05-01T00:01:00+08:00”>4294967295</value>
<value time=”2013-05-01T00:02:00+08:00”>0</value>

The maximum value of int type is 4294967295, and the minimum is 0. If it exceeds the boundary, unsignedLong or nonNegativeInteger type should be considered for use (see 6.2.10).

6.2.4 Decimal type

This standard should use Decimal type as `http://www.w3.org/2001/XMLSchema#decimal` defines (cited: decimal represents a subset of the real numbers, which can be represented by decimal numerals).

For example, the expressions in the content of value element are:

<value time=”2013-05-01T00:00:00+08:00”>826.1291</value>
<value time=”2013-05-01T00:01:00+08:00”>31239.46</value>
<value time=”2013-05-01T00:02:00+08:00”>1.02</value>
<value time=”2013-05-01T00:03:00+08:00”>-1.02</value>
<value time=”2013-05-01T00:04:00+08:00”>+10000.00</value>

NOTE—The type “decimal” is different from “float” type. For example, the value 32.8 cannot be expressed by floating value; because of the nature of floating point value type, the closest value to 32.8 is 32.799999 in floating type.

6.2.5 Float type

This standard should use Float type as `http://www.w3.org/2001/XMLSchema#float` defines.

For example, the expressions in the content of value element are:

<value time="2013-05-01T00:00:00+08:00">-1E4</value>
<value time="2013-05-01T00:01:00+08:00">1267.43233E12</value>
<value time="2013-05-01T00:02:00+08:00">12.78e-2</value>
<value time="2013-05-01T00:03:00+08:00">12</value>
<value time="2013-05-01T00:04:00+08:00">-0</value>
<value time="2013-05-01T00:05:00+08:00">0</value>
<value time="2013-05-01T00:06:00+08:00">INF</value>
<value time="2013-05-01T00:07:00+08:00">-INF</value>
<value time="2013-05-01T00:08:00+08:00">NaN</value>

NOTE—The type "float" is different from "decimal" type (see 6.2.4). Floating type should be used only if the values are managed as floating type at the binary level.

6.2.6 String type

This standard should use String type as `http://www.w3.org/2001/XMLSchema#string` defines.

For example, the expressions in the content of value element are:

<value time="2013-05-01T00:00:00+08:00">HEAT</value>
<value time="2013-05-01T00:01:00+08:00">COOL</value>
<value time="2013-05-01T00:02:00+08:00">AUTO</value>

6.2.7 Base64Binary type

This standard should use Base64Binary type as `http://www.w3.org/2001/XMLSchema#base64Binary` defines.

For example, the expression in the content of value element is:

<value time="2013-05-01T00:00:00+08:00">SUVFRTE4ODg=</value>

6.2.8 DateTime type

This standard should use DateTime type as `http://www.w3.org/2001/XMLSchema#dateTime` defines.

For example, the expression in the content of value element is:

<value time="2013-05-01T00:00:00+08:00">2013-05-01T12:00:00+08:00</value>

6.2.9 Duration type

This standard should use Duration type as `http://www.w3.org/2001/XMLSchema#duration` defines.

For example, the expressions in the content of value element are

```
<value time=”2013-05-01T00:00:00+08:00”> P1Y2M3DT10H30M</value>
<value time=”2013-05-01T00:01:00+08:00”> -P120D</value>
<value time=”2013-05-01T00:01:00+08:00”>0</value>
```

6.2.10 Other built-in data types

In the same manner, this standard allows other built-in data types. For example, to use very-large integer values, the point can use data type `http://www.w3.org/2001/XMLSchema#integer`, or `http://www.w3.org/2001/XMLSchema#double` data type to handle double-precise value. If it specifically uses “byte” data type, it can also use `http://www.w3.org/2001/XMLSchema#byte` data type.

6.3 Data type management using Registry

This subclause defines the rule to associate data type information to Point ID. The point element for registration and lookup can have “type” attribute defined in “http://ieee1888.org/1888.2/types” XML namespace.

This standard defines the XML namespace “http://ieee1888.org/1888.2/types”. This namespace has only “type” attribute with which the type of the value of Point can be explicitly described.

For example, to describe

http://example.org/POINT/01 is boolean type;
http://example.org/POINT/02 is int type;
http://example.org/POINT/03 is unsignedInt type;
http://example.org/POINT/04 is decimal type;
http://example.org/POINT/05 is string type;

the XML message for Point Registration request would be:

```
<transport   xmlns:xsd=”http://www.w3.org/2001/XMLSchema”
xmlns:n0=”http://ieee1888.org/1888.2/types”
… >
  <body>
<point id=”http://example.org/POINT/01” n0:type=”xsd:boolean” />
<point id=” http://example.org/POINT/02” n0:type=”xsd:int” />
<point id=” http://example.org/POINT/03” n0:type=”xsd:unsignedInt” />
<point id=” http://example.org/POINT/04” n0:type=”xsd:decimal” />
<point id=” http://example.org/POINT/05” n0:type=”xsd:string” />
  </body>
</transport>
```

Other data types defined under http://foreign.example.org/datatypeDefinition should be included in the following manner for import.

```
<transport   xmlns:foreign="http://foreign.example.org/datatypeDefinition"
xmlns:n0="http://ieee1888.org/1888.2/types"
… >
  <body>
<point id="http://example.org/POINT/08" n0:type="foreign:typeA" />
  </body>
</transport>
```

7. Importing field-bus data type

This clause defines the management rule for importing foreign data models mainly used in field-bus levels. Application protocols of field-buses usually have their own data definition (data model) and addresses on the network or on the memory space. The data definition sometimes not only specifies the primitive type of data but also the semantics of data. For example, in field-bus A, they may define that the values of type="HVAC_WIND_SPEED" are one of {"0", "1", "2", and "3"}. Here, they also define the meaning: e.g., "0" for stop, "3" for very high speed wind. This kind of definition is made outside the IEEE 1888 working group, but their data model should be also available in IEEE 1888.

The basic idea is to assign an XML namespace for the target data model. Then, design XML attributes under the XML namespace in order to put necessary parameters and information on Point ID. Then, such an attribute set can be electronically managed at the IEEE 1888 Registry.

This standard calls those organizations that import foreign data models "importers." Importers shall define an XML namespace for each foreign data model. The format of an XML namespace is open to them, but the IEEE 1888 working group uses the following URI for the prefix of the XML namespace: "http://ieee1888.org/1888.2/"

Let XML_NAMESPACE be the XML namespace that an Importer defined for a foreign data model. Under this XML_NAMESPACE, they shall define the names of attributes: ATTR0_NAME, ATTR1_NAME, ATTR2_NAME, and so on. Such attribute names shall have the type of the values, necessity (mandatory or optional), and the definition of the semantics. Annex C provides some examples.

With this two-phase definition, the following message can be sent to the Registry to associate the information of the foreign data model to the IEEE 1888 Point ID.

```
<transport xmlns:foreign="XML_NAMESPACE">
  <body>
    <point  id="http://example.org/Bldg1/001"  foreign:ATTR0_NAME="AI"  foreign:ATTR1_NAME="1"
/>
    <point  id="http://example.org/Bldg1/002"  foreign:ATTR0_NAME="AI"  foreign:ATTR1_NAME="2"
/>
    <point  id="http://example.org/Bldg1/003"  foreign:ATTR0_NAME="AI"  foreign:ATTR1_NAME="3"
/>
    <point  id="http://example.org/Bldg1/101"  foreign:ATTR0_NAME="AO"  foreign:ATTR1_NAME="1"
/>
    <point  id="http://example.org/Bldg1/102"  foreign:ATTR0_NAME="AO"  foreign:ATTR1_NAME="2"
/>
    <point  id="http://example.org/Bldg1/103"  foreign:ATTR0_NAME="AO"  foreign:ATTR1_NAME="3"
/>
  </body>
</transport>
```

NOTE—XML namespaces and associated attributes can be defined by any organizations. However, major (i.e., well-known) namespaces should be defined by some authorized organizations.[15]

8. Security considerations

The description in IEEE Std 1888™ Clause 10 shall apply. The standard recommends the considerations of 4.1 of IEEE Std 1888.3.

Since this standard introduces two new Components; RRS (5.2) and IAR (5.3), new threats should be considered. If a malicious IEEE 1888 application behaves as an RRS, it overwrites the mapping rule in the GW or the translation rule in an IAR. If a malicious IEEE 1888 application behaves as an IAR, it allows the malicious application to respond invalid values to a requester or to write invalid values to Providers. In order to mitigate those issues, this standard strongly recommends using IEEE Std 1888.3 in conjunction with this standard.

[15] Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement this standard.

Annex A

(informative)

ID mapping configuration between field-bus and IEEE 1888

This Annex shows examples of ID mapping configuration (see 5.2.2) in CSV format defined in RFC 4180 [B5]. In the CSV format, each line specifies configuration parameters including the definition of Point ID. The general format in the CSV file (this specification recommends to use) is `configuration_switch,...,...,...`

The "configuration_switch" is the parameter which defines the rule with the information in column 1, 2, 3, and so on. The configuration_switch can have following values:

- MODBUS_RTU: defines a binding between a Point ID and an entity in an modbus/rtu network
- MODBUS_RTU_DATATYPE: specifies a datatype on a modbus format
- MODBUS_TCP: defines a binding between a Point ID and an entity in a modbus/tcp network
- BACNET_IP: defines a binding between a Point ID and a parameter in a BACnet/IP network

The configuration switch defines how each Point ID is bound to the sensors or actuators in field-bus networks.

For MODBUS_RTU and MODBUS_RTU_DATATYPE, the format is:

```
MODBUS_RTU_DATATYPE, primitive_typename, modbus_endian
MODBUS_RTU, point_id, rtu_port_name, modbus_device_id,
modbus_register_number, modbus_datatype, readwrite_permission,
multiply_by, polling_interval
```

"primitive_typename" is the type name defined in XML Schema Part 2: Datatypes Second Edition.

"modbus_endian" specifies the memory order on the modbus register. The order should be specified with "->" from High byte to Low byte. For example, 0->1 means that the first byte (0) should be used at Higher byte, and the second byte (1) should be used at Lower byte.

(*) "MODBUS_RTU_DATATYPE,unsignedInt,2->3->0->1" means that if the data is presented as A1 B2 C3 D4 in hex format, the unsignedInt value should be:
0xC3*(2^24)+0xD4*(2^16)+0xA1*(2^8)+0xB2*(2^0).

"point_id" is a Point ID of IEEE 1888 (a URI that identifies the data in the IEEE 1888 network). With the following parameters, the binding between "point_id" and field-bus are specified.

"rtu_port_name" is a device name for modbus network (e.g., in Linux, it could be "/dev/ttyS0").

"modbus_device_id" is a device id in modbus network (e.g., 100).

"modbus_register_number" is a starting register number that specifies the object entity (e.g., 30).

"modbus_datatype" is the type name defined in XML Schema Data Type Definition, and it should be also declared in MODBUS_RTU_DATATYPE. This is a data type definition on modbus.

"readwrite_permission" is a permission setting for the specified modbus register address: "NONE" means no permission, "R" means read only, "W" means write only and "RW" means read write allowed.

“multiply_by” is a parameter that multiplies to the modbus values for showing it as an IEEE 1888 value, the data type is float (e.g., 32, 100, 1e+3, 0.001, -10, 1e-2).

“polling_interval” is an internal polling rate in second in modbus network for TRAP update. This is not intended for active polling and sending as a WRITE requester. If this parameter is disabled, 0 should be specified.

```
MODBUS_RTU_DATATYPE,int,0->1->2->3
MODBUS_RTU_DATATYPE,unsignedInt,0->1->2->3
MODBUS_RTU_DATATYPE,short,0->1
MODBUS_RTU_DATATYPE,unsignedShort,0->1

MODBUS_RTU,http://example.org/modbus/1/V,/dev/ttyS0,1,1,unsignedShort,R,1e-1,0
MODBUS_RTU,http://example.org/modbus/1/A,/dev/ttyS0,1,2,short,R,1e-1,0
MODBUS_RTU,http://example.org/modbus/1/KW,/dev/ttyS0,1,3,int,R,1e-3,0
MODBUS_RTU,http://example.org/modbus/1/KWH,/dev/ttyS0,1,5,unsignedInt,RW,1e-3,0
MODBUS_RTU,http://example.org/modbus/2/V,/dev/ttyS0,2,1,unsignedShort,R,1e-1,0
MODBUS_RTU,http://example.org/modbus/2/A,/dev/ttyS0,2,2,short,R,1e-1,0
MODBUS_RTU,http://example.org/modbus/2/KW,/dev/ttyS0,2,3,int,R,1e-3,0
MODBUS_RTU,http://example.org/modbus/2/KWH,/dev/ttyS0,2,5,unsignedInt,RW,1e-3,0
MODBUS_RTU,http://example.org/modbus/3/V,/dev/ttyS0,3,1,unsignedShort,R,1e-1,0
MODBUS_RTU,http://example.org/modbus/3/A,/dev/ttyS0,3,2,short,R,1e-1,0
MODBUS_RTU,http://example.org/modbus/3/KW,/dev/ttyS0,3,3,int,R,1e-3,0
MODBUS_RTU,http://example.org/modbus/3/KWH,/dev/ttyS0,3,5,unsignedInt,RW,1e-3,0
```

For BACNET_IP, the format is:

```
BACNET_IP, point_id, bacnetip_host, bacnetip_port, bacnetip_object_id,
bacnetip_parameter_id, bacnet_datatype, readwrite_permission,
multiply_by, polling_interval
```

“point_id” is a Point ID of IEEE 1888 (a URI that identifies the data in the IEEE 1888 network). With the following parameters, the binding between “point_id” and field-bus are specified.

“bacnetip_host” is a name or IP address of the BACnet/IP node (e.g., 192.168.11.102).

“bacnetip_port” is a UDP port number of the BACnet/IP node (e.g., 47808).

“bacnetip_object_id” is an object id of BACnet (e.g., 32, 0x000010).

“bacnetip_parameter_id” is a parameter id of the BACnet (e.g., 85, 0x55).

“bacnet_datatype” is a data type of BACnet: e.g., BOOLEAN, SIGNED, UNSIGNED, REAL, DOUBLE, OCTET_STRING, CHARACTER_STRING, BIT_STRING, ENUM, DATE, TIME, OBJECT_ID.

“readwrite_permission” is a permission setting for the specified BACnet object and parameter: “NONE” means no permission, “R” means read only, “W” means write only and “RW” means read write allowed.

“multiply_by” is a parameter that multiplies to the BACnet values for showing it as IEEE1888 value, the data type is float (e.g., 32, 100, 1e+3, 0.001, -10, 1e-2).

“polling_interval” is an internal polling rate in second in BACnet for TRAP update. This is not intended for active polling and sending as a WRITE requester. If this parameter is disabled, 0 should be specified.