INTERNATIONAL STANDARD

ISO/IEEE 11073-10419

Second edition
2019-03

Health informatics — Personal health device communication —

Part 10419: Device specialization — Insulin pump

Informatique de santé — Communication entre dispositifs de santé personnels —

Partie 10419: Spécialisation des dispositifs — Pompe à insuline

Reference number
ISO/IEEE 11073-10419:2019(E)

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

Email: stds.ipr@ieee.org

Published in Switzerland

Foreword

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

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of ISO documents should be noted (see www.iso.org/directives).

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.

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

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

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT), see www.iso.org/iso/foreword.html.

ISO/IEEE 11073-10419 was prepared by the IEEE 11073 Standards Committee of the IEEE Engineering in Medicine and Biology Society (as IEEE Std 11073-10419-2017) and drafted in accordance with its editorial rules. It was adopted, under the “fast-track procedure” defined in the Partner Standards Development Organization cooperation agreement between ISO and IEEE, by Technical Committee ISO/TC 215, *Health informatics*.

This second edition cancels and replaces the first edition (ISO 11073-10419:2016), which has been technically revised.

A list of all parts in the ISO 11073 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Abstract: Within the context of the ISO/IEEE 11073 family of standards for device communication, a normative definition of communication between personal telehealth insulin pump devices and compute engines (e.g., cell phones, personal computers, personal health appliances, set top boxes), in a manner that enables plug-and-play interoperability, is established in this standard. It leverages appropriate portions of existing standards including ISO/IEEE 11073 terminology, information models, application profile standards, and transport standards. It specifies the use of specific term codes, formats, and behaviors in telehealth environments restricting optionality in base frameworks in favor of interoperability. The standard defines a common core of communication functionality for personal telehealth insulin pump devices.

Keywords: IEEE 11073-10419™, insulin pump, medical device communication, personal health devices

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

PDF: ISBN 978-1-5044-4291-6 STD22758
Print: ISBN 978-1-5044-4292-3 STDPD22758

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. IEEE Standards 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 standard was submitted to the IEEE-SA Standards Board for approval, the Personal Health Devices Working Group had the following membership:

Daidi Zhong, *Chair*
Michael J. Kirwan, *Chair*
Melanie S. Yeung, *Vice Chair*
Akib Uddin, *Vice Chair*

Karsten Aalders
Charles R. Abbruscato
Nabil Abujbara
Maher Abuzaid
James Agnew
Haidar Ahmad
Manfred Aigner
Jorge Alberola
Murtaza Ali
Rolf Ambuehl
David Aparisi
Paolo Ariano
Lawrence Arne
Diego B. Arquillo
Serafin Arroyo
Muhammad Asim
Merat Bagha
Doug Baird
David Baker
Anindya Bakshi
Ananth Balasubramanian
Sunlee Bang
M. Jonathan Barkley
Gilberto Barrón
David Bean
John Bell
Rudy Belliardi
Kathryn M. Bennett
Daniel Bernstein
George A. Bertos
Chris Biernacki
Ola Björsne
Thomas Blackadar
Marc Blanchet
Thomas Bluethner
Douglas P. Bogia
Xavier Boniface
Shannon Boucousis
Julius Broma
Lyle G. Bullock
Bernard Burg
Chris Burns
Anthony Butt
Jeremy Byford-Rew
Satya Calloji
Xiaoying Cao
Carole C. Carey
Craig Carlson
Santiago Carot-Nemesio
Randy W. Carroll
Simon Carter
Seungchul Chae
Rahul Chauhan
James Cheng
Peggy Chien
David Chiu
Jinyong Choi
Chia-Chin Chong
Saeed A. Choudhary
Jinhan Chung
Malcolm Clarke
John A. Cogan
John T. Collins
Cory Condek
Todd H. Cooper
David Cornejo
Douglas Coup
Nigel Cox
Hans Crommenacker
Tomio Crosley
Allen Curtis
Ndifor Cyril Fru
Jesús Daniel Trigo
Eyal Dassau
David Davenport
Russell Davis
Sushil K. Deka
Ciro de la Vega
Pedro de-las-Heras-Quiros
Jim DelloStritto
Matthew d'Entremont
Kent Dicks
Hyoungho Do
Alistair Donaldson
Xiaolian Duan
Brian Dubreuil
Sourav Dutta
Jakob Ehrensvard
Fredrik Einberg
Michihiro Enokida
Javier Escayola Calvo
Mark Estes
Leonardo Estevez
Roger Feeley
Hailing Feng
Bosco T. Fernandes
Christoph Fischer
Morten Flintrup
Joseph W. Forler
Russell Foster
Eric Freudenthal
Matthias Frohner
Ken Fuchs
Jing Gao
Xuemei Gao
Marcus Garbe
John Garguilo
Rick Geimer
Igor Gejdos
Ferenc Gerbovics
Nicolae Goga
Julian Goldman
Raul Gonzalez Gomez
Chris Gough
Channa Gowda
Charles M. Gropper
Amit Gupta
Jeff Guttmacher
Rasmus Haahr
Christian Habermann
Michael Hagerty
Jerry Hahn
Robert Hall
Nathaniel Hamming
Rickey L. Hampton
Sten Hanke
Aki Harma
Jordan Hartmann
Kai Hassing
Marc Daniel Haunschild
Wolfgang Heck
Nathaniel Heintzman
Charles Henderson
Jun-Ho Her
Helen B. Hernandez
Takashi Hibino
Timothy L. Hirou
Allen Hobbs
Alex Holland
Arto Holopainen
Kris Holtzclaw
Robert Hoy
Frank Hsu
Anne Huang
Sen-Der Huang
Zhiqiang Huang
Ron Huby
David Hughes
Robert D. Hughes
Jiyoung Huh
Hugh Hunter
Hitoshi Ikeda
Yutaka Ikeda

Philip O. Isaacson
Atsushi Ito
Michael Jaffe
Praduman Jain
Wei Jin
Danny Jochelson
Phaneeth Junga
Akiyoshi Kabe
Steve Kahle
Tomio Kamioka
Kei Kariya
Andy Kaschl
Junzo Kashihara
Kohichi Kashiwagi
Ralph Kent
Laurie M. Kermes
Ikuo Keshi
Junhyung Kim
Minho Kim
Min-Joon Kim
Taekon Kim
Tetsuya Kimura
Alfred Kloos
Jeongmee Koh
Jean-Marc Koller
John Koon
Patty Krantz
Raymond Krasinski
Alexander Kraus
Ramesh Krishna
Geoffrey Kruse
Falko Kuester
Rafael Lajara
Pierre Landau
Jaechul Lee
JongMuk Lee
Kyong Ho Lee
Rami Lee
Sungkee Lee
Woojae Lee
Yonghee Lee
Joe Lenart
Kathryn A. Lesh
Catherine Li
Qiong Li
Patrick Lichter
Jisoon Lim
Joon-Ho Lim
John Lin
Wei-Jung Lo
Charles Lowe
Don Ludolph
Christian Luszick
Bob MacWilliams
Srikkanth Madhurbootheswaran
Miriam L. Makhlouf
Romain Marmot
Sandra Martinez
Miguel Martínez de Espronceda Cámara
Peter Mayhew
Jim McCain
László Meleg
Alexander Mense
Jinsei Miyazaki
Erik Moll
Darr Moore
Carsten Mueglitz
Piotr Murawski
Soundharya Nagasubramanian
Jae-Wook Nah
Alex Neefus
Trong-Nghia Nguyen-Dobinsky
Michael E. Nidd
Tetsu Nishimura
Jim Niswander
Hiroaki Niwamoto
Thomas Norgall
Anand Noubade
Yoshiteru Nozoe
Abraham Ofek
Brett Olive
Begonya Otal
Marco Paleari
Charles Palmer
Bud Panjwani
Carl Pantiskas
Harry P. Pappas
Hanna Park
Jong-Tae Park
Myungeun Park
Soojun Park
Phillip E. Pash
TongBi Pei
Lucian Pestritu
Soren Petersen
James Petisce
Peter Piction
Michael Pliskin
Varshney Prabodh
Jeff Price
Harald Prinzhorn
Harry Qiu
Arif Rahman
Tanzilur Rahman
Steve Ray
Phillip Raymond
Tim Reilly
Barry Reinhold
Brian Reinhold
Melvin I. Reynolds
John G. Rhoads
Jeffrey S. Robbins
Chris Roberts
Moskowitz Robert
Timothy Robertson
David Rosales
Fatemeh Saki
Bill Saltzstein
Benedikt Salzbrunn
Giovanna Sannino
Jose A. Santos-Cadenas
Stefan Sauermann
John Sawyer
Guillaume Schatz
Alois Schloegl
Paul S. Schluter
Lars Schmitt
Mark G. Schnell
Richard A. Schrenker
Antonio Scorpiniti
Kwang Seok Seo
Riccardo Serafin
Sid Shaw
Frank Shen
Bozhi Shi
Min Shih
Mazen Shihabi
Redmond Shouldice
Sternly K. Simon
Marjorie Skubic
Robert Smith
Ivan Soh
Motoki Sone
Emily Sopensky
Rajagopalan Srinivasan
Andreas Staubert
Nicholas Steblay
Lars Steubesand
John (Ivo) Stivoric
Raymond A. Strickland
Chandrasekaran Subramaniam
Hermanni Suominen
Lee Surprenant
Ravi Swami
Ray Sweidan
Jin Tan
Yi Tang
Haruyuyki Tatsumi
John W. Thomas
Jonas Tirén
Alexandra Todiruta
Janet Traub
Gary Tschautscher
Masato Tsuchid
Ken Tubman
Yoshihiro Uchida
Sunil Unadkat
Fabio Urbani
Philipp Urbauer
Laura Vanzago
Alpo Värri
Dalimar Velez
Rudi Voon
Barry Vornbrock
Isobel Walker
David Wang
Jerry P. Wang
Yao Wang
Yi Wang
Steve Warren
Fujio Watanabe
Toru Watsuji
Mike Weng
Kathleen Wible
Paul Williamson
Jan Wittenber
Jia-Rong Wu
Will Wykeham
Ariton Xhafa
Dan Xiao
Yaxi Yan

Qifeng Yan
Junjie Yang
Ricky Yang
Qiang Yin
Done-Sik Yoo
Jianchao Zeng
Jason Zhang
Zhiqiang Zhang
Thomas Zhao
Miha Zoubek
Szymon Zyskoter

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

John Ballingall
Hector Barron Gonzalez
Lyle G. Bullock
Keith Chow
Joseph El Youssef
Randall Groves
Kai Hassing
Werner Hoelzl
Noriyuki Ikeuchi
Atsushi Ito
Piotr Karocki
Patrick Keith-Hynes
Patrick Kinney
Robert Kircher
Michael J. Kirwan
Nick S. A. Nikjoo
Henry Pinto
Melvin I. Reynolds
Bartien Sayogo
Lars Schmitt
Raymond Strickland
Walter Struppler
Jan Wittenber
Oren Yuen
Daidi Zhong

When the IEEE-SA Standards Board approved this standard on 28 September 2017, it had the following membership:

Jean-Philippe Faure, *Chair*
Gary Hoffman, *Vice Chair*
John D. Kulick, *Past Chair*
Konstantinos Karachalios, *Secretary*

Chuck Adams
Masayuki Ariyoshi
Ted Burse
Stephen Dukes
Doug Edwards
J. Travis Griffith
Michael Janezic
Thomas Koshy
Joseph L. Koepfinger*
Kevin Lu
Daleep Mohla
Damir Novosel
Ronald C. Petersen
Annette D. Reilly
Robby Robson
Dorothy Stanley
Adrian Stephens
Mehmet Ulema
Phil Wennblom
Howard Wolfman
Yu Yuan

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 11073-10419-2017, Health informatics—Personal health device communication—Part 10419: Device Specialization—Insulin Pump.

ISO/IEEE 11073 standards enable communication between medical devices and external computer systems. This document uses the optimized framework created in ISO/IEEE 11073-20601:2016 and describes a specific, interoperable communication approach for insulin pumps.[1] These standards align with, and draw on, the existing clinically focused standards to provide support for communication of data from clinical or personal health devices (PHDs).

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

Contents

Health informatics—Personal health device communication

Part 10419: Device Specialization—Insulin Pump

1. Overview

1.1 Scope

This standard establishes a normative definition of communication between personal telehealth insulin pump devices (agents) and managers (e.g., cell phones, personal computers, personal health appliances, set top boxes) in a manner that enables plug-and-play interoperability. It leverages work done in other ISO/IEEE 11073 standards including existing terminology, information profiles, application profile standards, and transport standards. It specifies the use of specific term codes, formats, and behaviors in telehealth environments, restricting optionality in base frameworks in favor of interoperability. This standard defines a common core functionality of personal telehealth insulin pump devices.

In the context of personal health devices (PHDs), an insulin pump is a medical device used for the administration of insulin in the treatment of diabetes mellitus, also known as continuous subcutaneous insulin infusion (CSII) therapy.

This standard provides the data modeling according to ISO/IEEE 11073-20601 and does not specify the measurement method.

1.2 Purpose

This standard addresses the need for an openly defined, independent standard that supports information exchange to and from PHDs and compute engines (e.g., cell phones, personal computers, personal health appliances, set top boxes). Interoperability is key to growing the potential market for these devices and to enabling people to be better informed participants in the management of their health.

1.3 Context

See ISO/IEEE 11073-20601:2016 for an overview of the environment within which this standard is written.[2]

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

This standard defines the device specialization for the insulin pump, being a specific agent type, and provides a description of the device concepts, its capabilities, and its implementation according to this standard.

This standard is based on ISO/IEEE 11073-20601:2016, which in turn draw information from both ISO/IEEE 11073-10201:2004 [B8] and ISO/IEEE 11073-20101:2004 [B9].[3] The medical device encoding rules (MDER) used within this standard are fully described in ISO/IEEE 11073-20601:2016.

This standard reproduces relevant portions of the nomenclature found in ISO/IEEE 11073-10101:2004 [B6] and ISO/IEEE 11073-10101a:2015 [B7] and adds new nomenclature codes for the purposes of this standard. Among these standards and ISO/IEEE 11073-20601:2016, all required nomenclature codes for implementation are documented.

NOTE—In this standard, ISO/IEEE 11073-104zz is used to refer to the collection of device specialization standards that utilize ISO/IEEE 11073-20601, where zz can be any number from 01 to 99, inclusive.[4]

2. Normative references

The following referenced documents are indispensable for the application of this document (i.e., they must be understood and used; therefore, 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.

ISO/IEEE 11073-20601:2016, Health informatics—Personal health device communication—Part 20601: Application Profile—Optimized Exchange Protocol.[5]

See Annex A for all informative material referenced by this standard.

3. Definitions, acronyms, and abbreviations

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.[6]

agent: A node that collects and transmits personal health data to an associated manager.

artificial pancreas: A system combining diabetes devices to provide similar functionality as a pancreas. Examples include linking a continuous glucose monitor to an insulin pump to automatically reduce or increase insulin infusion based upon specified thresholds of measured interstitial glucose.

basal insulin: Insulin required to cover the basic insulin needs of the body.

basal rate: Rate of continuously delivered insulin to cover the basic insulin needs of the body.

[3] The numbers in brackets correspond to the numbers of the bibliography in Annex A.
[4] Notes in text, tables, and figures of a standard are given for information only and do not contain requirements needed to implement the standard.
[5] ISO/IEEE publications are available from the ISO Central Secretariat (http://www.iso.ch/). ISO/IEEE publications are also available in the United States from The Institute of Electrical and Electronics Engineers (http://standards.ieee.org/).
[6] *IEEE Standards Dictionary Online* is available at http://dictionary.ieee.org.

basal rate profile: Profile defining the time-dependent course of the basal rate over a periodic time interval (e.g., 24 hours, beginning at midnight).

blood glucose: Glucose concentration in the blood.

bolus: A single dose of a medication typically administered by infusion. *See also:* **bolus insulin**.

bolus insulin: Insulin required to cover the intake of food or to correct high blood glucose levels.

class: In object-oriented modeling, a term collectively describing the attributes, methods, and events utilized by objects instantiated from the class.

compute engine: *See:* **manager**.

correction bolus: A bolus to correct high blood glucose levels. *See also:* **bolus**.

daily basal dose: Total amount of basal insulin delivered in 24 hours, beginning at midnight.

device: A physical apparatus implementing either an agent or a manager role.

extended bolus: A bolus given over an extended period of time with a specified duration. *See also:* **bolus**.

fast bolus: A bolus given over a very short period that can be considered as having infinitesimal duration and specified by amount of infusion. *See also:* **bolus**.

glucagon: Naturally occurring hormone released by the pancreas when glucose levels are low.

glucose: The major source of energy used by the body cells. Glucose is commonly referred to as *sugar*.

handle: An unsigned 16-bit number that is locally unique and identifies one of the object instances within an agent.

infusion set: A tubing system with cannula that interfaces the insulin reservoir to the body.

insulin: Naturally occurring hormone required for carbohydrate metabolism. Insulin regulates the way sugar, fat, and proteins are moved into the cells and the way they are stored or used for energy.

insulin-to-carbohydrate ratio (I:CHO): Amount of carbohydrate covered or disposed of by a single unit of insulin.

insulin sensitivity factor (ISF): An estimated amount of blood glucose reduced by a single unit of insulin. *Syn:* **correction factor**.

international unit (IU): Unit of measurement used in pharmacology for the measurement of the amount of a substance.

manager: A node receiving data from one or more agent systems. Some examples of managers include a cellular phone, health appliance, set top box, or computer system.

meal bolus: A bolus to compensate for meal intake. *See also:* **bolus**.

multi-wave bolus: A bolus combining a fast bolus with an extended bolus. One part of the bolus amount is delivered immediately while the other is delivered over a specific period of time.

object: In object-oriented modeling, a particular instantiation of a class. The instantiation realizes attributes, methods, and events from the class.

obj-handle: *See:* **handle**.

occlusion: Total obstruction of the infusion set that prevents administering insulin.

personal health device (PHD): A device used in personal health applications.

personal telehealth device: *See:* **personal health device (PHD)**.

piston rod: Mechanical apparatus to propel the plunger of the insulin reservoir.

priming: The filling of the fluidic path with insulin (e.g., after replacement of the reservoir and/or infusion set).

reservoir: The insulin supply of an insulin pump.

temporary basal rate: Temporary adjustment of the basal rate.

total daily dose: The total amount of insulin delivered during one day (i.e., 24 hours, beginning at midnight). This amount contains basal insulin as well as bolus insulin.

3.2 Acronyms and abbreviations

AP	artificial pancreas
APDU	application protocol data unit
ASN.1	Abstract Syntax Notation One
CSII	continuous subcutaneous insulin infusion, commonly known as *insulin pump therapy*
DIM	domain information model
DM	disease management
EUI-64	extended unique identifier (64 bits)
HCP	health care professional
I:CHO	insulin-to-carbohydrate ratio
ICS	implementation conformance statement
ID	identifier
ISF	insulin sensitivity factor
IU	international unit
MDC	medical device communication
MDER	medical device encoding rules
MDS	medical device system
MOC	managed object class
OID	object identifier
PDU	protocol data unit
PHD	personal health device
VMO	virtual medical object
VMS	virtual medical system

4. Introduction to ISO/IEEE 11073 personal health devices (PHDs)

4.1 General

This standard and the remainder of the series of ISO/IEEE 11073 PHD standards fit in the larger context of the ISO/IEEE 11073 series of standards. The full suite of standards enables agents to interconnect and interoperate with managers and with computerized health-care information systems. See ISO/IEEE 11073-20601:2016 for a description of the guiding principles for this series of ISO/IEEE 11073 PHD standards.

ISO/IEEE 11073-20601:2016 supports the modeling and implementation of an extensive set of PHDs. This standard defines aspects of the insulin pump device. It describes all aspects necessary to implement the application layer services and data exchange protocol between an ISO/IEEE 11073 PHD insulin pump agent and a manager. This standard defines a subset of the objects and functionality contained in ISO/IEEE 11073-20601:2016 and extends and adds definitions where appropriate. All new definitions are given in Annex B in Abstract Syntax Notation One (ASN.1) (ITU-T X.680-2002 [B10]). Nomenclature codes referenced in this standard that are not defined in ISO/IEEE 11073-20601:2016 are normatively defined in Annex C.

4.2 Introduction to ISO/IEEE 11073-20601 modeling constructs

4.2.1 General

The ISO/IEEE 11073 series of standards, and in particular ISO/IEEE 11073-20601:2016, is based on an object-oriented systems management paradigm. The overall system model is divided into three principal components: the domain information model (DIM), the service model, and the communication model. See ISO/IEEE 11073-20601:2016 for a detailed description of the modeling constructs.

4.2.2 Domain information model (DIM)

The DIM is a hierarchical model that describes an agent as a set of objects. These objects and their attributes represent the elements that control behavior and report on the status of the agent and data that an agent can communicate to a manager. Communication between the agent and the manager is defined by the application protocol in the ISO/IEEE 11073-20601:2016.

4.2.3 Service model

The service model defines the conceptual mechanisms for the data exchange services. Such services are mapped to messages that are exchanged between the agent and the manager. Protocol messages within the ISO/IEEE 11073 series of standards are defined in ASN.1. The messages defined in ISO/IEEE 11073-20601:2016 can coexist with messages defined in other standard application profiles defined in the ISO/IEEE 11073 series of standards.

4.2.4 Communication model

In general, the communication model supports the topology of one or more agents communicating over logical point-to-point connections to a single manager. For each logical point-to-point connection, the dynamic system behavior is defined by a connection state machine as specified in ISO/IEEE 11073-20601:2016.

4.2.5 Implementing the models

An agent implementing this standard shall implement all mandatory elements of the information, service, and communication models as well as all conditional elements where the condition is met. The agent should implement the recommended elements, and it may implement any combination of the optional elements. A manager implementing this standard shall utilize at least one of the mandatory, conditional, recommended, or optional elements. In this context, *utilize* means to use the element as part of the primary function of the manager device. For example, a manager whose primary function is to display data would need to display a piece of data in the element in order to utilize it.

4.3 Compliance with other standards

Devices that comply with this standard may also be required to comply with other domain-and device-specific standards that supersede the requirements of this standard with respect to issues including safety, reliability, and risk management. A user of this standard is expected to be familiar with all other such standards that apply and to comply with any higher specifications thus imposed. Typically, medical devices should comply with the IEC 60601-1:2005 [B1] base standards with respect to electrical and mechanical safety and any device-specific standard as might be defined in the IEC 60601-2 [B2] series of standards. Software aspects may apply through standards such as IEC 62304:2006/EN 62304:2006 [B3].

Devices that comply with this standard implement higher layers of network software and utilize lower layers as appropriate to the application. The requirements on performance of such applications and conformance are defined elsewhere and are outside the scope of this standard. Moreover, the use of any medical equipment is subject to risk assessment and risk management appropriate to the application. Some relevant examples are ISO 14971:2007 [B5] and IEC 80001-1:2010 [B4]. The requirements of such risk assessment and risk management and conformance are outside the scope of this standard.

5. Insulin pump device concepts and modalities

5.1 General

This clause presents the general concepts of insulin pump devices. In the context of PHDs in this family of standards, an insulin pump is a device that administers insulin at a defined amount to the human body. Insulin pump devices are primarily used in the continuous subcutaneous insulin infusion (CSII) therapy of type 1 diabetes mellitus. This type of diabetes mellitus is characterized by loss of the insulin-producing beta cells of the islets of Langerhans in the pancreas. Insulin pumps typically inject insulin into the subcutaneous layer of fat tissue under the skin through an infusion set. Preferred sites for the cannula are the abdomen, lumbar region, thighs, buttocks, and the upper arms.

Insulin is a hormone that affects the metabolism and other body functions. It regulates, together with glucagon, the glucose level in the blood. Failure to produce sufficient insulin to maintain the blood glucose in the normal range leads to conditions of temporarily or persistently high blood sugar (hyperglycemia).

The objective of CSII therapy, also known as insulin pump therapy, is to mimic the insulin secretion of the healthy body. Insulin pump therapy therefore differentiates between basal insulin and bolus insulin. Basal insulin, sometimes also referred to as *background* insulin, regulates the blood glucose level without taking into account any food intake. In the context of the CSII therapy, basal insulin is continuously administered at a given rate depending on the fluctuating need throughout the day. By contrast, bolus insulin compensates for an increased need for insulin induced by food intake. This is also known as *meal* bolus, and the insulin dosage can be determined with the knowledge of the user's insulin-to-carbohydrate ratio (I:CHO) (the amount of carbohydrate covered by a unit of insulin). In addition, bolus insulin may also be administered to decrease a temporarily high blood glucose level. This is often referred to as *correction* bolus. Meal or correction boluses can be delivered as a standard fast bolus, extended bolus, or as a multi-

wave bolus. Fast bolus is an infusion of insulin over a very short period similar to a pen injection that can be considered as having infinitesimal duration. An extended bolus is an infusion of insulin over a longer period extending the insulin delivery. A multi-wave bolus combines a fast bolus with an extended bolus; one part of the bolus amount is delivered immediately while the other is delivered over a specific period of time. This type of bolus is suitable for meals that contain both fast and slowly digestible carbohydrates or long meals with several courses.

Although the distinction between basal and bolus insulin has a high relevance for the CSII therapy, the insulin pump typically delivers a single insulin type for both delivery methods. Unlike the intensified conventional therapy (ICT, injection therapy, pen therapy) where typically a combination of injections with fast-acting and slow-acting insulin is utilized, the insulin pump therapy generally uses rapid- or fast-acting insulin. Another term for these types of insulin is *insulin analog*.

Many insulin pump devices allow the pump user to program basal rate profiles that define the fluctuating delivery of basal insulin of the pump in the course of the day. These profile settings are often referred to as circadian profiles. Well-adjusted basal rate profiles generally cover the insulin pump user's need for basal insulin and there is normally no necessity to adjust these profiles on a daily basis. However, there are situations in which the need for basal insulin deviates at such a rate that a temporary adjustment of the basal delivery is indicated. Phases of stress, intensified physical activity, illness, and illnesses are counted among the reasons for increased or reduced need for basal insulin. The temporary basal rate is a means to make short-term adjustments to the basal insulin delivery. Two different concepts for the temporary basal rate are currently well accepted in the area of insulin pump therapy. The first concept applies a scaling factor to the programmed basal rate for a certain period of time. The second concept consists in overriding the programmed basal rate with a specific value for a certain time period.

The unit of measurement to define the dose of insulin is international unit (IU). One IU of insulin is the biological equivalent of 1/22 mg of pure crystalline insulin. The concentration of insulin in an insulin solution is defined by international units per milliliter (i.e., IU/mL). In pharmacology, it is also very common to indicate the concentration of insulin solutions by the letter U followed by the number of international units per milliliter. For example, a concentration of 100 IU/mL is denominated as U100.

5.2 Device types

Insulin pump devices are generally designed to be portable and permanently connected to the body.

The structural shape of insulin pumps may vary, but insulin pump devices typically include the following components: the pump unit, a disposable insulin reservoir, and a disposable infusion set.

The pump unit typically comprises a computing unit and a dosing mechanism, both powered by a battery. Additionally, pump units may contain several user interface elements such as displays, buttons, or apparatuses to generate visual, haptic, or acoustic signals.

The disposable insulin reservoir typically consists of a glass or plastic ampulla with a volumetric capacity of a few milliliters of insulin solution. Depending on the insulin need of the pump user, this capacity is sufficient for provisioning insulin for several days.

The disposable infusion set interfaces the insulin reservoir to the human body. It generally comprises an adapter that connects to the insulin reservoir, a tubing system and a cannula that is inserted into the subcutaneous fat tissue. It is generally recommended to replace the infusion set after 2 to 3 days of usage for sterility reasons and to help prevent irritation at the insertion site.

5.3 Collected data

5.3.1 General

The insulin pump is a portable device and therefore may not be connected to a manager while collecting data. The two main use cases for an insulin pump agent to connect to a manager and send its data are the following:

— The insulin pump user visits a health care professional (HCP) to examine the adequacy of the therapy. The HCP normally compares the historic data from the insulin pump with the corresponding data from a blood glucose meter or continuous glucose monitor device to derive necessary adjustments to the insulin pump therapy. As the interval between such visits may constitute several months, insulin pumps are typically capable of storing data for such time periods.

— The insulin pump user connects the insulin pump agent to a manager at home to examine the adequacy of the therapy and to apply adjustments when indicated. This tends to happen on a more frequent basis (e.g., once per week).

In addition to the two use cases above, an insulin pump agent may also be continuously connected to a manager to report collected data (e.g., artificial pancreas).

5.3.2 Current bolus setting

The current bolus setting represents the amount of insulin requested to be delivered as a bolus and is reported before the pump delivers the bolus. A bolus is typically initiated by the insulin pump user to compensate for an increased need for insulin induced by intake of food containing a relatively high quantity of carbohydrates (e.g., meal bolus). A bolus may also be indicated to decrease a temporarily high blood glucose level (e.g., correction bolus).

The quantity of insulin specified in a current bolus setting can be delivered in a theoretically infinitesimal period of time (e.g., fast bolus) or can be delivered over a defined period of time (e.g., extended bolus) or as a combination of a fast bolus with an extended bolus (e.g., multi-wave bolus). The duration of an extended bolus shall be reported as a measure active period.

A current bolus setting can be set to be delivered at a later time. In this instance, the current bolus setting shall be used with the pending bolus delay together to specify the amount of delay between programming a current bolus setting and the expected bolus delivered.

A current bolus setting can be cancelled. In this instance, the current bolus setting shall be used with the insulin pump status to specify the bolus canceled.

A bolus amount can be set manually by the user, or recommended by the insulin pump's bolus calculator, or commanded from an artificial pancreas controller. If an insulin pump device has a bolus calculator, the user may accept or change the recommended amount of insulin.

The unit for the bolus setting is international units.

NOTE—This object represents a device setting and not a measurement value.

5.3.3 Pending bolus delay

The pending bolus delay specifies the delay between programming a current bolus setting and the expected bolus delivered. This setting is helpful for patients with delayed digestion (gastroparesis).

The unit for the pending bolus delay is in minutes.

NOTE—This object represents a device setting and not a measurement value.

5.3.4 Bolus delivered

The bolus delivered measurement represents the amount of insulin effectively delivered by the insulin pump as a bolus and is reported after the pump has delivered the bolus.

The measurement represents the delivered amount of insulin for a bolus and the modality can be described as a fast bolus, extended bolus, or multi-wave bolus, and the reason for delivering the bolus as meal bolus or correction bolus (Figure 1). The unit for the bolus delivered is international units.

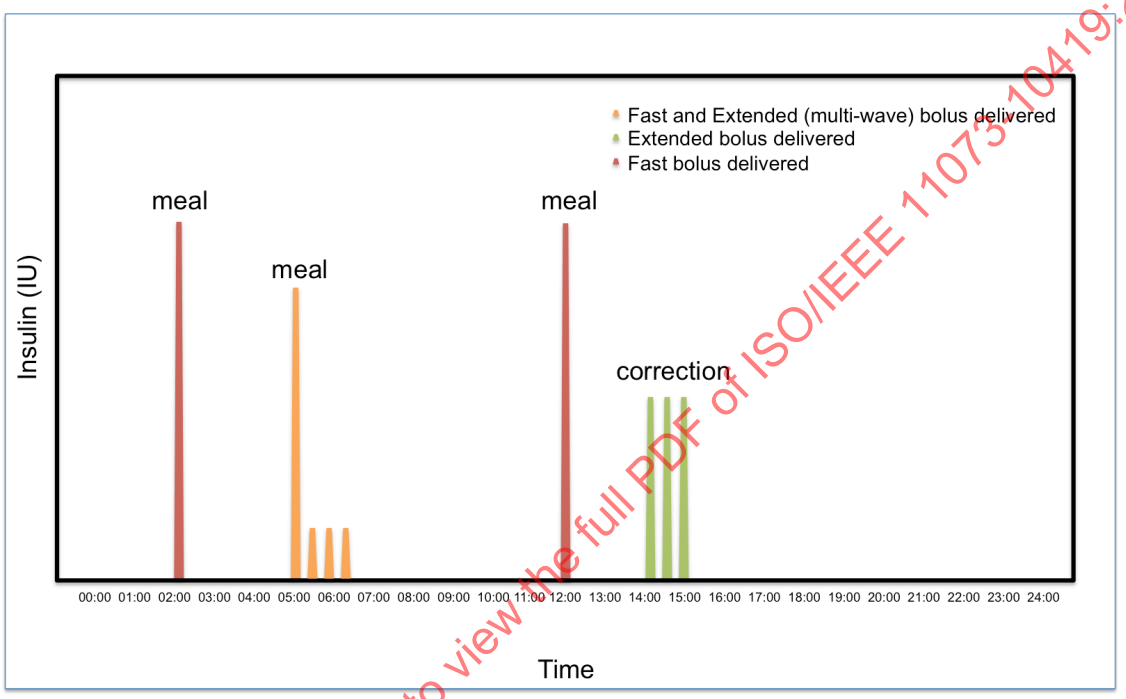

Figure 1—Example of bolus delivered

5.3.5 Basal rate schedule setting

The purpose of a basal rate schedule setting is to define the amount of basal insulin programmed in an insulin pump to be delivered for a specified schedule according to the individually fluctuating physiological needs. One or more basal rate schedule settings form a 24 hour basal profile.

The unit of a basal rate schedule setting is international units per hour.

NOTE—This object represents a device setting and not a measurement value.

5.3.6 Basal profile setting

Insulin pump devices typically offer the option of programming multiple basal profile settings in order to easily meet changing insulin needs (e.g., work day versus weekend).

In the context of CSII, a basal rate designates the amount of basal insulin programmed to deliver over a specified period of time.

The basal profile setting is a schedule-segment that consists of a series of basal rate schedule settings programmed on the insulin pump device. The basal profile setting has a schedule that typically begins at 12 a.m. and ends at 12 a.m. the following day accumulating a total duration of 24 hours (Figure 2).

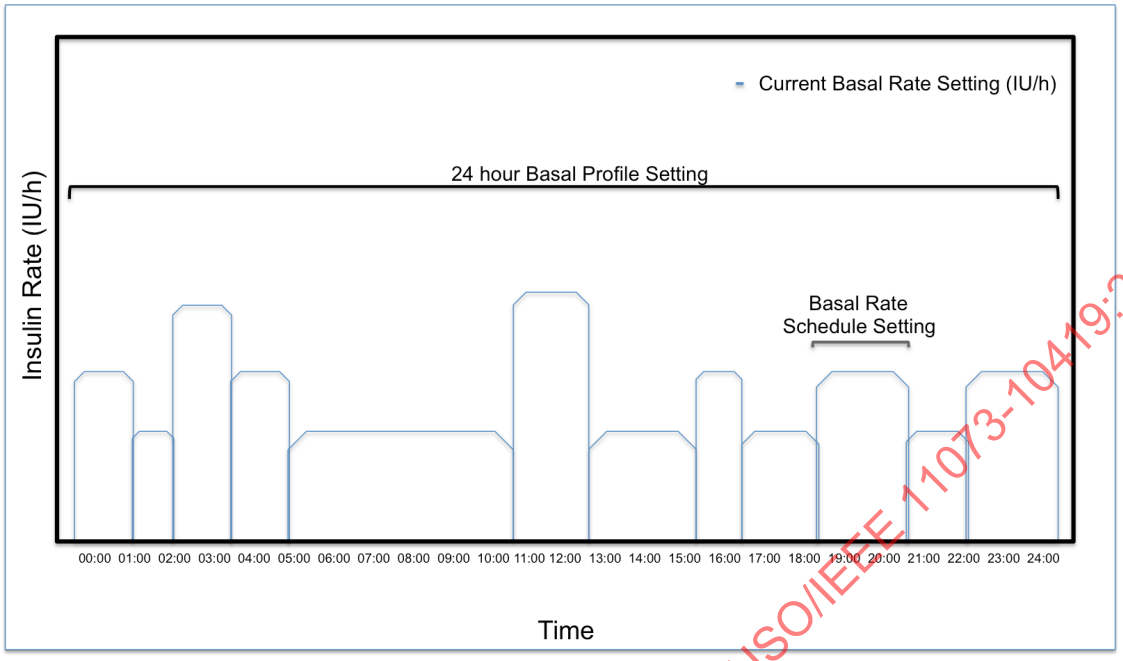

Figure 2—Example of basal profile setting consisting of basal rate schedule settings

Insulin pump devices typically offer the option of programming multiple basal profile settings in order to easily meet changing insulin needs (e.g., work day versus weekend). The basal profile setting will also have a unique identifier specified by the manufacturer to identify a basal profile setting.

If an insulin pump deviace supports programming basal profile settings, an active basal profile setting can be selected.

NOTE—This object represents a device setting and not a measurement value.

5.3.7 Current basal rate setting

The current basal rate setting represents the most recent basal rate to be delivered by the pump and is reported before the pump delivers the basal rate. Current basal rate setting may be initiated by a programmed basal profile or a temporary basal rate adjustment made by the user.

A basal profile is a scheduled sequence of basal rate schedule settings programmed on the insulin device. The active basal profile and current time may define the current basal rate setting.

Alternatively, a user may initiate a temporary basal rate adjustment as a result of a temporarily changed need. This adjustment remains active for a defined period of time and afterwards the current basal rate setting will revert to the programmed basal profile. The temporary basal rate setting can be initiated by a *relative* or an *absolute* adjustment.

A *relative* temporary basal rate setting applies a dimensionless scaling factor to the current basal rate for the active basal profile. Therefore if the relative temporary basal rate is active, the rate delivery equals the value of the reference factor multiplied by the basal rates given by the active basal profile for a given period of time. A relative temporary basal rate shall be the calculated rate after applying the scaling factor.

An *absolute* temporary basal rate setting overrides the basal rates given by the active basal rate profile for a given period of time. Therefore, if the absolute temporary basal rate is active, the rate delivery equals the value of the absolute temporary basal rate for a given period of time.

This standard assumes that the effective delivery of basal insulin can be represented or approximated by discrete values of constant delivery rate over time. This standard assumes that the temporary basal rate setting overrides the basal rates given by the active basal rate profile. Therefore, if the temporary basal rate (either relative or absolute) is active, the rate of the actual basal insulin delivery equals the value of the temporary basal rate. The unit of the delivery rate is international units per hour.

NOTE—This object represents a device setting and not a measurement value.

5.3.8 Basal delivered

The basal delivered measurement represents the amount of insulin effectively delivered as a basal by an insulin pump and is reported after the pump has delivered the basal. An insulin pump device commonly utilizes semi-continuous infusion and incrementally injects discrete amounts (often called a *micro-bolus*) of insulin to satisfy the current basal rate setting (Figure 3). The insulin pump is responsible for determining the frequency of the basal amount to be reported (e.g., insulin pump may report the amount of insulin delivered at the end of a basal rate or intermittently as micro-boluses).

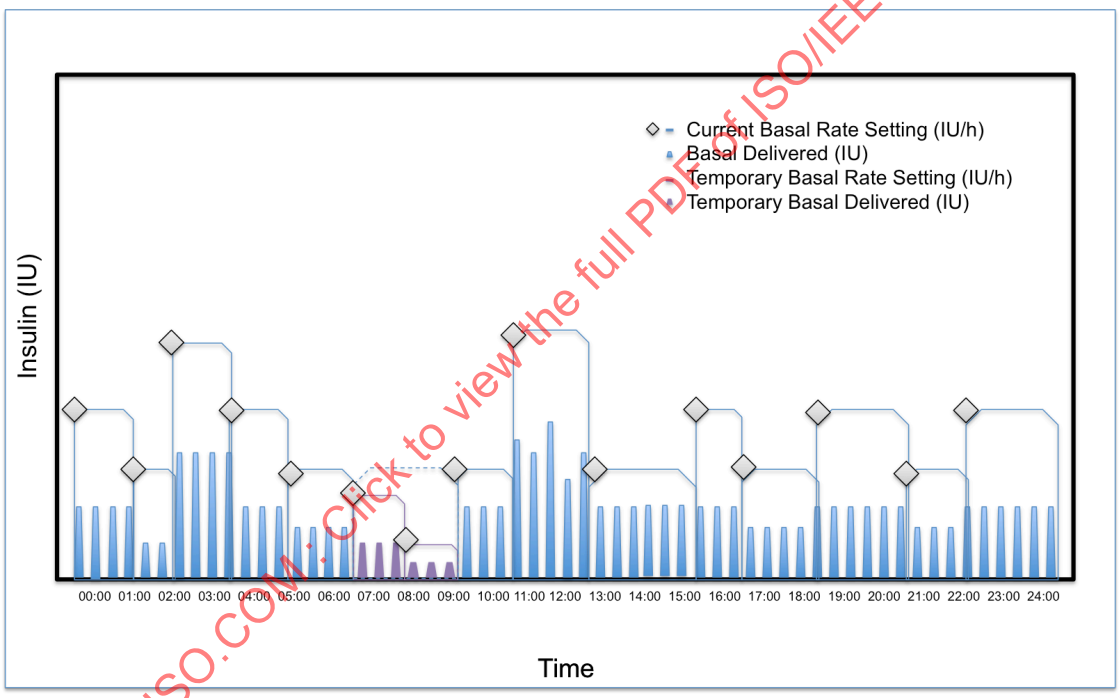

Figure 3—Example of basal rate setting and basal delivered

The measurement represents the delivered amount of basal insulin and can be described as a programmed basal rate, basal rate set by a controller, a temporary absolute basal, temporary relative basal, or undetermined basal rate. The unit for the basal delivered measurement is international units.

5.3.9 Insulin reservoir remaining

The insulin reservoir remaining measurement represents the amount of insulin remaining in the reservoir. The reservoir holds insulin inside the insulin pump and can have varying capacity. As the pump delivers insulin, the insulin amount in the reservoir decreases. When the reservoir is empty, users typically refill and insert a new reservoir into the insulin pump. The insulin pump is responsible for determining the frequency of the insulin reservoir remaining amount to be reported (e.g., the insulin reservoir remaining may be reported at the same time as the bolus-delivered or basal-delivered measurements).

The measurement represents the amount of insulin remaining in the reservoir. The unit for the insulin reservoir remaining is international units.

5.3.10 Insulin concentration

The insulin concentration measurement represents the concentration of insulin in the reservoir. Normally it is specified as the amount of insulin in international units per milliliter of insulin solution. For example, U100 designates an insulin concentration of 100 IU/mL of insulin solution.

The measurement represents the insulin concentration. The unit for the insulin concentration is international units per milliliter.

5.3.11 Insulin-to-carbohydrate ratio (I:CHO) profile settings

This represents the patient's I:CHO, which is the amount of carbohydrate covered or disposed of by a 1 IU of insulin.

Insulin pump devices may offer the option of storing multiple I:CHOs as a patient may have varying ratios specific to a time of day or type of meal. The unit for the I:CHO profile setting is grams.

NOTE—This object represents a device setting and not a measurement value.

5.3.12 Insulin sensitivity factor (ISF) profile settings

This represents the patient's ISF or sometimes referred to as a *correction factor*, which is an estimated amount of blood glucose reduced by a single unit of insulin.

Insulin pump devices may offer the option of storing multiple ISFs as a patient may have varying factors specific to the time of day. The unit for the ISF profile setting is millimole/liter or milligrams/deciliter.

NOTE—This object represents a device setting and not a measurement value.

5.3.13 Operational status

The *operational status* represents the operation and therapy conditions of the insulin pump device. Operational conditions are Off, Stand-By, Preparing, Priming, Waiting, Ready, and Undetermined. Therapy conditions are Pause, Stop, and Run.

5.3.14 PHD DM status

The *PHD DM status* allows generic notification handling for personal health devices. The status of the device is represented in a number of bit flags covering warning, error, service, and undetermined statuses. Changes in status are sent as events notifying the manager.

This represents a generic notification mechanism for a device to report warnings, errors, and other abnormal statuses to a manager that may provide visibility of these status change events to the user.

5.3.15 Insulin pump status

The *insulin pump status* represents the specific notifications given by the insulin pump for handling, warnings, errors, reminders, or confirmations to user-initiated events meant to provide visibility of events to the user and manufacture.

5.4 Stored data

As stated in 5.3.1, an insulin pump may be used over several months of operation without being connected to a manager to send its data. Once an insulin pump is connected to a manager, the manager is able to select which of the agent's stored measurements or observations to retrieve. Depending on the agent's capabilities to organize its data into clusters of chronologically contiguous data, the manager may also select the time ranges of the stored data to retrieve. The agent then transmits the manager's selection in one or several blocks of messages for processing by a manager or other processing apparatus. The manager may also be able to choose a set of data clusters for deletion.

5.5 Scheduled data

An insulin pump may have the ability to program scheduled events such as basal profile settings, I:CHO profile settings, and ISF profile settings. These scheduled data provide information about the actions or events and time points that the actions or events occur. Once an insulin pump is connected to a manager, the manager is able to select which of the agent's scheduled data to retrieve. The agent then transmits the manager's selection in one or several blocks of messages for processing by a manager or other processing apparatus. The manager may also be able to choose a set of data clusters for deletion.

6. Insulin pump domain information model (DIM)

6.1 Overview

This clause describes the DIM of the insulin pump.

6.2 Class extensions

In this standard, extensions are defined with respect to ISO/IEEE 11073-20601:2016.

The following four attributes are an extension to the enumeration class.

— The Capability-Mask-Simple and Capability-Mask-Basic attributes indicate whether the corresponding bit in the Enum-Observed-Value-Simple-Bit-Str or Enum-Observed-Value-Basic-Bit-Str attribute, respectively, is supported by the Agent.

— The State-Flag-Simple and State-Flag-Basic attributes indicate whether the corresponding bit in the Enum-Observed-Value-Simple-Bit-Str or Enum-Observed-Value-Basic-Bit-Str attribute, respectively, is a state or event.

If the Enum-Observed-Value-Simple-Bit-Str attribute is used, the Capability-Mask-Simple and State-Flag-Simple attributes are mandatory. If the Enum-Observed-Value-Basic-Bit-Str attribute is used, the Capability-Mask-Basic and State-Flag-Basic attributes are mandatory. The values of the Capability-Mask-Simple, Capability-Mask-Basic, State-Flag-Simple, and State-Flag-Basic attributes are implementation specific.

See Annex B for their ASN.1 structures.

The Capability-Mask-Simple or Capability-Mask-Basic attribute is dynamic, i.e., the attribute may change at some point after configuration. The State-Flag-Simple or State-Flag-Basic attribute is static, i.e., the attribute shall remain unchanged after the configuration is agreed upon.

6.3 Object instance diagram

The metric object instance diagram of the insulin pump DIM, defined for the purposes of this standard, is shown in Figure 4.

The objects of the DIM, as shown in Figure 4, include the medical device system (MDS) object (see 6.6), the numeric objects (see 6.7), the real-time sample array objects (see 6.8), the enumeration objects (see 6.9), the persistent metric–store (PM-store) objects (see 6.10), and the scanner objects (see 6.11). See 6.14 for rules for extending the insulin pump information model beyond elements as described in this standard. Each subclause that describes an object of the insulin pump contains the following information:

— The nomenclature code used to identify the class of the object. One example where this code is used is the configuration event, where the object class is reported for each object. This allows the manager to determine whether the class of the object being specified is a numeric, real-time sample array, enumeration, scanner, or PM-store class.

— The attributes of the object. Each object has attributes that represent and convey information on the physical device and its data sources. Each object has a handle attribute that identifies the object instance within an agent. Attribute values are accessed and modified using methods such as GET and SET. Attribute types are defined using ASN.1. The ASN.1 definitions for new attribute types specific to this standard are in Annex B, and the ASN.1 definitions for existing attribute types referenced in this standard are in ISO/IEEE 11073-20601:2016.

— The methods available on the object.

— The potential events generated by the object. The data are sent to the manager using events.

— The available services such as getting or setting attributes.

The attributes for each class are defined in tables that specify the name of the attribute, its value, and its qualifier. The qualifiers are defined as follows:

— M: attribute is mandatory.

— C: attribute is conditional and depends on the condition stated in the Remark or Value column (if ISO/IEEE 11073-20601:2016 is referenced, then it contains the conditions).

— R: attribute is recommended.

— NR: attribute is not recommended.

— O: attribute is optional.

Mandatory attributes shall be implemented by an agent. Conditional attributes shall be implemented if the condition applies and may be implemented otherwise. Recommended attributes should be implemented by the agent. Not recommended attributes should not be implemented by the agent. Optional attributes may be implemented by the agent. For attributes with qualifiers set to R or NR, underlying requirements stated in the Remark and Value column in ISO/IEEE 11073-20601:2016 shall be followed.

The attributes can be either static, i.e., they shall remain unchanged after the configuration is agreed upon, or dynamic, i.e., the attribute may change at some point after configuration.

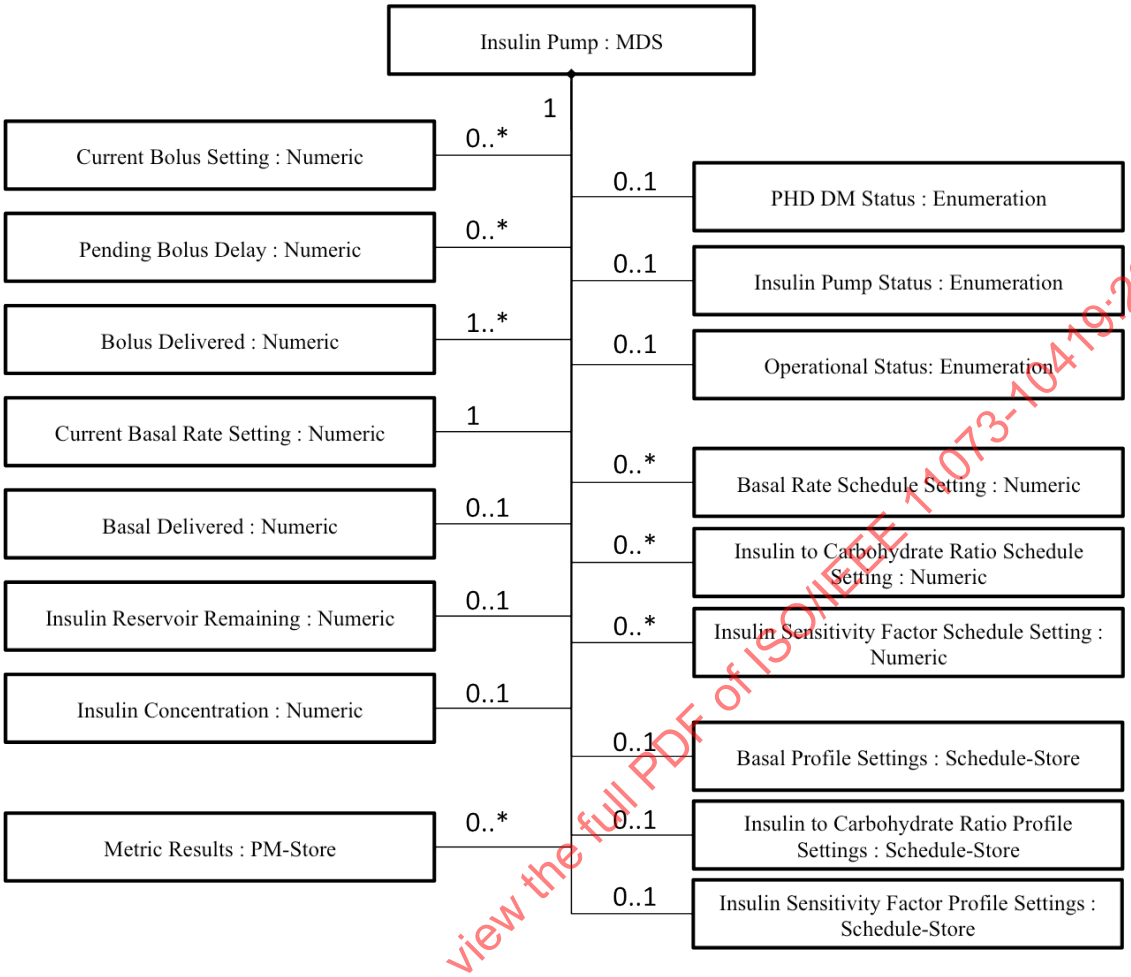

Figure 4—Insulin pump DIM

6.4 Types of configuration

6.4.1 General

As specified in ISO/IEEE 11073-20601:2016, there are two styles of configuration available. Subclauses 6.4.2 and 6.4.3 briefly introduce standard and extended configurations.

6.4.2 Standard configuration

Standard configurations are defined in the ISO/IEEE 11073-104zz specializations (such as this standard) and are assigned a well-known identifier (Dev-Configuration-Id). The usage of a standard configuration is negotiated at association time between the agent and the manager. If the manager recognizes and selects to operate using the configuration, then the agent can send measurements immediately. If the manager does not recognize the configuration, the agent provides the configuration prior to transmitting measurement information.

One standard configuration is defined in this standard. The standard configuration 1900 (0x076C) contains one bolus delivered object (see 6.7.4) and one current basal rate setting object (see 6.7.5).

6.4.3 Extended configuration

In extended configurations, the agent's configuration is not predefined in a standard. The agent determines the objects, attributes, and values that to be used in a configuration and assigns a configuration identifier. When the agent associates with a manager, an acceptable configuration is negotiated. Typically, the manager does not recognize the agent's configuration on the first connection, so the manager responds that the agent needs to send its configuration information as a configuration event report. If, however, the manager recognizes the configuration, either because it was preloaded in some way or the agent had previously associated with the manager, then the manager responds that the configuration is known and no further configuration information needs to be sent.

6.5 Profiles

6.5.1 General

A profile further constrains the objects, services, and communication model of a specialization. By profiling the device specialization, the standard provides more guidance on the specific mandatory objects that shall be implemented, the objects that are optional, and the objects that are not required. This standard does not define profiles for the insulin pump device.

6.6 MDS object

6.6.1 MDS object attributes

Table 1 summarizes the attributes of the insulin pump MDS object. The nomenclature code to identify the MDS class is MDC_MOC_VMS_MDS_SIMP.

Table 1—MDS object attributes

Attribute name	Value	Qualifier
Handle	0	M
System-Type	Attribute not present. See ISO/IEEE 11073-20601:2016.	C
System-Type-Spec-List	{MDC_DEV_SPEC_PROFILE_INSULIN_PUMP, 2}.	M
System-Model	{"Manufacturer","Model"}	M
System-Id	Extended unique identifier (64-bits) (EUI-64)	M
Dev-Configuration-Id	Standard config: 0x076C (1900) Extended configs: 0x4000-0x7FFF	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Production-Specification	See ISO/IEEE 11073-20601:2016.	O
Mds-Time-Info	See ISO/IEEE 11073-20601:2016.	C
Date-and-Time	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time	See ISO/IEEE 11073-20601:2016.	M
Relative-Time	See ISO/IEEE 11073-20601:2016.	C
HiRes-Relative-Time	See ISO/IEEE 11073-20601:2016.	C
Date-and-Time-Adjustment	See ISO/IEEE 11073-20601:2016.	C
Power-Status	See ISO/IEEE 11073-20601:2016.	R
Battery-Level	See ISO/IEEE 11073-20601:2016.	R
Remaining-Battery-Time	See ISO/IEEE 11073-20601:2016.	R
Reg-Cert-Data-List	See ISO/IEEE 11073-20601:2016.	O
Confirm-Timeout	See ISO/IEEE 11073-20601:2016.	O

NOTE—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

In the response to a Get MDS object command, only implemented attributes and their corresponding values are returned.

See ISO/IEEE 11073-20601:2016 for descriptive explanations of the individual attributes as well as for information on attribute ID and attribute type.

The Dev-Configuration-Id attribute holds a locally unique 16-bit identifier that identifies the device configuration. For an insulin pump agent with extended configuration, this identifier is chosen in the range of extended-config-start to extended-config-end (see ISO/IEEE 11073-20601:2016) as shown in Table 1.

The agent sends the Dev-Configuration-Id during the Associating state (see 8.3) to identify its configuration for the duration of the association. If the manager already holds the configuration information relating to the Dev-Configuration-Id, it recognizes the Dev-Configuration-Id and the Configuring state (see 8.4) is skipped, and the agent and manager then enter the Operating state. If the manager does not recognize the Dev-Configuration-Id, the agent and manager enter the Configuring state.

If an agent implements multiple ISO/IEEE 11073-104zz specializations, System-Type-Spec-List is a list of type/version pairs, each referencing the respective device specialization and version of that specialization.

6.6.2 MDS object methods

Table 2 defines the methods (actions) of the MDS object. These methods are invoked using the Action service. In Table 2, the Subservice type name column defines the name of the method; the Mode column defines whether the method is invoked as an unconfirmed action (i.e., roiv-cmip-action from ISO/IEEE 11073-20601:2016) or a confirmed action (i.e., roiv-cmip-confirmed-action); the Subservice type (action-type) column defines the nomenclature code to use in the action-type field of an action request and response (see ISO/IEEE 11073-20601:2016); the Parameters (action-info-args) column defines the associated ASN.1 data structure (see ISO/IEEE 11073-20601:2016 for ASN.1 definitions) to use in the action message for the action-info-args field of the request; and the Results (action-info-args) column defines the structure to use in the action-info-args of the response.

Table 2—MDS object methods

Service	Subservice type name	Mode	Subservice type (action-type)	Parameters (action-info-args)	Results (action-info-args)
ACTION	Set-Time	Confirmed	MDC_ACT_SET_TIME	SetTimeInvoke	—
	Set-Base-Offset-Time	Confirmed	MDC_ACT_SET_BO_TIME	SetBOTimeInvoke	—

— **Set-Time:**

This method allows the manager to set a real-time clock in the agent with the absolute time. The agent indicates whether the Set-Time command is valid using the mds-time-capab-set-clock bit in the Mds-Time-Info attribute (see ISO/IEEE 11073-20601:2016).

If the agent supports the Absolute-Time-Stamp attribute, this method shall be implemented.

— **Set-Base-Offset-Time:**

This method allows the manager to set a real-time clock in the agent with the base time and offset. The agent indicates whether the Set-Base-Offset-Time command is valid using the mds-time-capab-set-clock bit in the Mds-Time-Info attribute (see ISO/IEEE 11073-20601:2016).

If the agent supports the Base-Offset-Time-Stamp attribute, this method shall be implemented.

Agents following only this device specialization and no others shall send event reports using agent-initiated measurement data transmission. Agents following this device specialization as well as others shall send

event reports in the appropriate fashion. During the association procedure (see 8.3), data-req-mode-capab shall be set to the appropriate value for the event report style. As a result, the manager shall assume the insulin pump agent does not support any of the MDS-Data-Request features (see ISO/IEEE 11073-20601:2016 for additional information). Thus, implementation of the MDS-Data-Request method/action is not required in this standard and is not shown in Table 2.

6.6.3 MDS object events

Table 3 defines the events that can be sent by the insulin pump MDS object.

Table 3—Insulin pump MDS object events

Service	Subservice type name	Mode	Subservice type (event-type)	Parameters (event-info)	Results (event-reply-info)
EVENT REPORT	MDS-Configuration-Event	Confirmed	MDC_NOTI_CONFIG	ConfigReport	ConfigReport Rsp
	MDS-Dynamic-Data-Update-Var	Confirmed	MDC_NOTI_SCAN_REPORT_VAR	ScanReportInfoVar	—
	MDS-Dynamic-Data-Update-Fixed	Confirmed	MDC_NOTI_SCAN_REPORT_FIXED	ScanReportInfo Fixed	—
	MDS-Dynamic-Data-Update-MP-Var	Confirmed	MDC_NOTI_SCAN_REPORT_MP_VAR	ScanReportInfoMP Var	—
	MDS-Dynamic-Data-Update-MP-Fixed	Confirmed	MDC_NOTI_SCAN_REPORT_MP_FIXED	ScanReportInfoMP Fixed	—

— **MDS-Configuration-Event:**

This event is sent by the insulin pump agent during the configuring procedure if the manager does not already know the insulin pump agent's configuration from past associations or because the manager has not been implemented to recognize the configuration according to the insulin pump device specialization. The event provides static information about the supported measurement capabilities of the insulin pump agent.

— **MDS-Dynamic-Data-Update-Var:**

This event provides dynamic measurement data from the insulin pump agent for the numeric and enumeration objects. These data are reported using a generic attribute list variable format. The event is sent as an unsolicited message by the agent (i.e., an agent-initiated measurement data transmission). See 8.5.3 for more information on unsolicited event reporting.

— **MDS-Dynamic-Data-Update-Fixed:**

This event provides dynamic measurement data from the insulin pump agent for the numeric and enumeration objects. These data are reported in the fixed format defined by the Attribute-Value-Map attribute of the objects. The event is sent as an unsolicited message by the agent (i.e., an agent-initiated measurement data transmission). See 8.5.3 for more information on unsolicited event reporting.

— **MDS-Dynamic-Data-Update-MP-Var:**

This is the same as MDS-Dynamic-Data-Update-Var but allows inclusion of data from multiple people.

— **MDS-Dynamic-Data-Update-MP-Fixed:**

This is the same as MDS-Dynamic-Data-Update-Fixed but allows inclusion of data from multiple people.

NOTE—ISO/IEEE 11073-20601:2016 requires that managers support all of the MDS Object Events listed above.

6.6.4 Other MDS services

6.6.4.1 GET service

An insulin pump agent shall support the GET service, which is provided by the MDS object to retrieve the values of all implemented MDS object attributes. The GET service can be invoked as soon as the insulin pump agent receives the Association Response and moves to the Associated state, including the Operating and Configuring substates.

The manager may request the MDS object attributes of the insulin pump agent; in which case, the manager shall send the "Remote Operation Invoke | Get" message (see roiv-cmip-get in ISO/IEEE 11073-20601:2016) with the reserved MDS handle value of 0. The insulin pump agent shall report its MDS object attributes to the manager using the "Remote Operation Response | Get" message (see rors-cmip-get in ISO/IEEE 11073-20601:2016). See Table 4 for a summary of the GET service including some message fields.

Table 4—Insulin pump MDS object GET service

Service	Subservice type name	Mode	Subservice type	Parameters	Results
GET	<na>	<implied confirmed>	<na>	GetArgumentSimple = (obj-handle = 0), attribute-id-list <*optional*>	GetResultSimple = (obj-handle = 0), attribute-list

See 8.5.2 for details on the procedure for getting the MDS object attributes.

6.6.4.2 SET service

The insulin pump specialization does not require an implementation to support the MDS object SET service.

6.7 Numeric objects

6.7.1 General

The insulin pump DIM for metric objects (see Figure 4) contains numeric objects that represent aspects of insulin delivery.

Sometimes, the interpretation of one attribute value in an object depends on other attribute values in the same object. For example, Unit-Code and Unit-LabelString provide context for the observed values. Whenever a contextual attribute changes, the agent shall report these changes to the manager using an MDS object event (see 6.6.3) prior to reporting any of the dependent values.

6.7.2 Current bolus setting

Table 5 summarizes the attributes of the current bolus setting numeric object. The nomenclature code to identify the numeric class is MDC_MOC_VMO_METRIC_NU. The current bolus setting numeric object may be present in the extended configuration.

Table 5—Current bolus setting numeric object attributes

Attribute name	Extended configuration	
	Value	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	{MDC_PART_PHD_DM | MDC_INS_BOLUS_SET}	M
Supplemental-Types	See following text in Table 6.	R
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-acc-manager-initiated | mss-acc-agent-initiated | mss-cat-manual | mss-cat-setting | mss-cat-calculation	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	O
Measurement-Status	See ISO/IEEE 11073-20601:2016.	O
Metric-Id	See following text in Table 7.	R
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	O
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR
Unit-Code	MDC_DIM_INTL_UNIT	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See ISO/IEEE 11073-20601:2016.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	O
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See following text.	C
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	NR
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	R
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The value reported in this object is the amount of insulin requested to be delivered as a bolus and is reported before the pump delivers the bolus. Only non-negative numbers shall be used.

An insulin pump agent may contain multiple bolus setting objects if it is designed to transmit multiple modalities of bolus delivery, in which case the modality shall be denoted in the Supplemental-Type attribute.

The Supplemental-Types attribute is used to distinguish the modality and the reason for delivering a bolus of insulin. Table 6 represents the possible values to express a fast or extended bolus delivery mode as well as correction or meal bolus delivery reason.

Certain combinations of Supplemental-Types are allowed. If it is desired to express that a meal bolus was delivered as a fast bolus, the SupplementalTypesList structure of the Supplemental-Types attribute should contain the two values MDC_INS_BOLUS_MEAL and MDC_INS_BOLUS_FAST. Similarly, a correction bolus delivered as an extended bolus should contain MDC_INS_BOLUS_CORR and MDC_INS_BOLUS_EXT in the SupplementalTypesList.

The modality values MDC_INS_BOLUS_FAST and MDC_INS_BOLUS_EXT shall not be combined in the SupplementalTypesList describing the bolus context. However, a multi-wave bolus modality may be represented by reporting separate current bolus setting objects with MDC_INS_BOLUS_FAST and MDC_INS_BOLUS_EXT at the same time.

If an acceptable, existing nomenclature term is not available, MDC_INS_BOLUS_UNDETERMINED shall be used.

Table 6—Bolus context

Type	Definition
MDC_INS_BOLUS_FAST	Bolus insulin delivered as fast bolus
MDC_INS_BOLUS_EXT	Bolus insulin delivered as an extended bolus
MDC_INS_BOLUS_CORR	Bolus insulin delivered for a correction
MDC_INS_BOLUS_MEAL	Bolus insulin delivered for a meal
MDC_INS_BOLUS_UNDETERMINED	Undetermined bolus

The Metric-ID attribute is used to distinguish the bolus setting source and described in Table 7.

If an acceptable, existing nomenclature term is not available, MDC_INS_BOLUS_OTHER shall be used.

Table 7—Bolus source

Type	Definition
MDC_INS_BOLUS_MANUAL	Manual, user defined bolus
MDC_INS_BOLUS_RECOMMENDED	Recommended bolus (e.g., bolus calculator)
MDC_INS_BOLUS_MANUAL_CHANGE	Recommended bolus changed by a user
MDC_INS_BOLUS_COMMANDED	Commanded bolus (e.g., artificial pancreas controller)
MDC_INS_BOLUS_OTHER	Other source of bolus

The Measure-Active-Period attribute shall be used with the MDC_INS_BOLUS_EXT to describe the time duration of the extended bolus setting.

The current bolus setting numeric object does not support any methods, events, or other services.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as for information on attribute ID and attribute type.

6.7.3 Pending bolus delay

Table 8 summarizes the attributes of the pending bolus delay numeric object. The nomenclature code to identify the numeric class is MDC_MOC_VMO_METRIC_NU. The pending bolus delay numeric object may be present in the extended configuration.

Table 8—Pending bolus delay numeric object attributes

Attribute name	Extended configuration	
	Value	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	{MDC_PART_PHD_DM | MDC_INS_BOLUS_PENDING_DELAY}	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	NR
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-acc-manager-initiated | mss-acc-agent-initiated | mss-cat-manual | mss-cat-setting	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	O
Measurement-Status	See ISO/IEEE 11073-20601:2016.	O
Metric-Id	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR
Unit-Code	MDC_DIM_MIN or MDC_DIM_SEC	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See following text.	R
Label-String	See ISO/IEEE 11073-20601:2016.	O
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Simple-Nu-Observed-Value	See /ISO/IEEE 11073-20601:2016.	C
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	M
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The value reported in this object is the amount of time between programming a bolus and the expected bolus delivered. Only non-negative numbers shall be used.

If the Source-Handle-Reference is defined, it should point to the current bolus setting object.

The pending bolus delay numeric object does not support any methods, events, or other services.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as for information on attribute ID and attribute type.

6.7.4 Bolus delivered

Table 9 summarizes the attributes of the bolus delivered numeric object. The nomenclature code to identify the numeric class is MDC_MOC_VMO_METRIC_NU. The bolus delivered numeric object shall be supported by an insulin pump agent.

Table 9—Bolus delivered numeric object attributes

Attribute name	Extended configuration		Standard configuration (Dev-Configuration-Id = 0x076C)	
	Value	Qualifier	Value	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M	1	M
Type	{MDC_PART_PHD_DM | MDC_INS_BOLUS}	M	{MDC_PART_PHD_DM | MDC_INS_BOLUS}	M
Supplemental-Types	See following text in Table 10.	R	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016. See following text in Table 10.	O
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-manager-initiated | mss-acc-agent-initiated | mss-cat-calculation	M	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-manager-initiated | mss-acc-agent-initiated | mss-cat-calculation	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	O	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	O
Measurement-Status	See ISO/IEEE 11073-20601:2016.	O	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	O
Metric-Id	See ISO/IEEE 11073-20601:2016. See following text.	O	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016. See following text.	O
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	O	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Unit-Code	MDC_DIM_INTL_UNIT	M	MDC_DIM_INTL_UNIT	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C	MDC_ATTR_NU_VAL_OBS_BASIC, then MDC_ATTR_TIME_STAMP_BO	M
Source-Handle-Reference	See ISO/IEEE 11073-20601:2016. See following text.	R	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	O
Label-String	See ISO/IEEE 11073-20601:2016.	O	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR

(Table continues on the next page.)

Table 9—Bolus delivered numeric object attributes *(continued)*

Attribute name	Extended configuration		Standard configuration (Dev-Configuration-Id = 0x076C)	
	Value	Qualifier	Value	Qualifier
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R	If fixed format is used and the standard configuration is not adjusted, this attribute is mandatory; otherwise, the conditions from ISO/IEEE 11073-20601a-2010 apply.	M
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Compound-Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016	C
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	M	If fixed format is used and the standard configuration is unchanged, this attribute is mandatory; otherwise, the conditions from ISO/IEEE 11073-20601:2016 apply.	M
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

For an insulin pump agent with standard configuration the AttrValMap structure (see ISO/IEEE 11073-20601:2016) of the Attribute-Value-Map attribute shall contain the attribute ID and attribute length information of the Basic-Nu-Observed-Value and Base-Offset-Time-Stamp attribute in the same order as indicated in Table 9.

The observed value reported in this object is the amount of the insulin effectively delivered as a bolus and is reported after the pump has delivered the bolus. Only non-negative numbers shall be used.

The Supplemental-Types attribute is used to distinguish the modality of a particular bolus and the reason for delivering a bolus of insulin. Table 10 represents the possible values to express a fast or extended bolus delivery mode as well as correction or meal bolus delivery reason.

Table 10—Bolus context

Type	Definition
MDC_INS_BOLUS_FAST	Bolus insulin delivered as fast bolus
MDC_INS_BOLUS_EXT	Bolus insulin delivered as an extended bolus
MDC_INS_BOLUS_CORR	Bolus insulin delivered for a correction
MDC_INS_BOLUS_MEAL	Bolus insulin delivered for a meal
MDC_INS_BOLUS_UNDETERMINED	Undetermined bolus

Certain combinations of Supplemental-Types are allowed. If it is desired to express that a meal bolus was delivered as a fast bolus, the SupplementalTypesList structure of the Supplemental-Types attribute should contain the two values MDC_INS_BOLUS_MEAL and MDC_INS_BOLUS_FAST. Similarly, a correction bolus delivered as an extended bolus should contain MDC_INS_BOLUS_CORR and MDC_INS_BOLUS_EXT in the SupplementalTypesList.

The modality values MDC_INS_BOLUS_FAST and MDC_INS_BOLUS_EXT shall not be combined in the SupplementalTypesList describing the bolus context. However, a multi-wave bolus modality may be represented by reporting separate bolus objects with MDC_INS_BOLUS_FAST and MDC_INS_BOLUS_EXT at the same time.

If an acceptable, existing nomenclature term is not available, MDC_INS_BOLUS_UNDETERMINED shall be used.

If the Source-Handle-Reference is defined, it should point to the current bolus setting object.

The bolus delivered numeric object does not support any methods, events, or other services.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as for information on attribute ID and attribute type.

6.7.5 Current basal rate setting

Table 11 summarizes the attributes of the current basal rate setting numeric object. The nomenclature code to identify the numeric class is MDC_MOC_VMO_METRIC_NU. The current basal rate setting numeric object shall be supported by an insulin pump agent.

Table 11—Current basal rate setting numeric object attributes

Attribute name	Extended configuration		Standard configuration (Dev-Configuration-Id = 0x076C)	
	Value	Qualifier	Value	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M	2	M
Type	{MDC_PART_PHD_DM | MDC_INS_BASAL_RATE_SET}	M	{MDC_PART_PHD_DM | MDC_INS_BASAL_RATE_SET}	M
Supplemental-Types	See following text in Table 12	M	MDC_INS_BASAL_PRGM	M
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-acc-manager-initiated | mss-acc-agent-initiated | mss-cat-setting	M	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-acc-manager-initiated | mss-acc-agent-initiated | mss-cat-setting	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See following text in Table 13.	O	MDC_INS_BASAL_DEVICE initially. For other possible values that could be later reported, see following text in Table 13.	M
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Unit-Code	MDC_DIM_INTL_UNIT_PER_HR	M	MDC_DIM_INTL_UNIT_PER_HR	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C	MDC_ATTR_NU_VAL_OBS_BASIC, then MDC_ATTR_TIME_STAMP_BO.	M
Source-Handle-Reference	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	O	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C

(Table continues on the next page.)

Table 11—Current basal rate setting numeric object attributes *(continued)*

Attribute name	Extended configuration		Standard configuration (Dev-Configuration-Id = 0x076C)	
	Value	Qualifier	Value	Qualifier
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R	If fixed format is used and the standard configuration is unchanged, this attribute is mandatory; otherwise, the conditions from ISO/IEEE 11073-20601:2016 apply.	M
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Compound-Simple-Nu-Observed-Value	See /ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016	C
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	R	If fixed format is used and the standard configuration is unchanged, this attribute is mandatory; otherwise, the conditions from ISO/IEEE 11073-20601:2016 apply.	M
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR	Attribute not initially present. If present follow ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

For an insulin pump agent with standard configuration the AttrValMap structure (see ISO/IEEE 11073-20601:2016) of the Attribute-Value-Map attribute shall contain the attribute ID and attribute length information of the Basic-Nu-Observed-Value and Base-Offset-Time-Stamp attribute in the same order as indicated in Table 11.

The value reported in this object is the rate of insulin requested to be delivered as a basal and is reported prior to the delivery of the basal. Only non-negative numbers shall be used.

The Supplemental-Types attribute is used to distinguish additional context of a particular basal rate. Table 12 represents the possible values to express a programmed, temporary absolute or temporary relative basal rate.

If an acceptable, existing nomenclature term is not available, MDC_INS_BASAL_UNDETERMINED shall be used.

Table 12—Basal rate context

Type	Definition
MDC_INS_BASAL_PRGM	Rate set by the active basal profile schedule setting
MDC_INS_BASAL_TEMP_ABS	Rate set by an absolute temporary change
MDC_INS_BASAL_TEMP_REL	Rate set by a relative temporary change
MDC_INS_BASAL_UNDETERMINED	Undetermined basal rate context

The Metric-Id attribute is used to distinguish the source of a particular basal rate setting. Table 13 represents the possible values to express a basal rate setting by the device, a remote or an artificial pancreas controller.

Table 13—Basal rate source

Type	Definition
MDC_INS_BASAL_DEVICE	Rate set by the insulin pump
MDC_INS_BASAL_REMOTE	Rate set by a remote controller
MDC_INS_BASAL_AP_CTRL	Rate set by an artificial pancreas controller
MDC_INS_BASAL_OTHER	Rate set by an other source

The current basal rate setting numeric object does not support any methods, events, or other services.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as for information on attribute ID and attribute type.

6.7.6 Basal delivered

Table 14 defines the attributes for the object that represents the basal delivered. The nomenclature code to identify the numeric class is MDC_MOC_VMO_METRIC_NU. The basal delivered object may be supported by an insulin pump agent with extended configuration.

Table 14—Basal delivered numeric object attributes

Attribute name	Extended configuration	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	{MDC_PART_PHD_DM| MDC_INS_BASAL}	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	O
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-manager-initated | mss-acc-agent-initiated | mss-cat-calculation	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See ISO/IEEE 11073-20601:2016.	O
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	C
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	O
Unit-Code	MDC_DIM_INTL_UNIT	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See ISO/IEEE 11073-20601:2016.	O
Label-String	See ISO/IEEE 11073-20601:2016.	O
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	R
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The observed value reported in this object is the amount of the insulin effectively delivered as a basal and is reported after the pump has delivered the basal. The insulin pump is responsible for determining the frequency of the basal delivered event to be reported (i.e., insulin pump may report the amount of insulin delivered at the end of a current basal rate setting or intermittently as micro-boluses). Only non-negative numbers shall be used.

The basal delivered numeric object does not support any methods, events, or other services.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as information on attribute ID and attribute type.

6.7.7 Basal rate schedule setting

Table 15 defines the attributes for the numeric object that represents the basal rate for the schedule.

The basal rate schedule setting object may be supported by an insulin pump agent with extended configuration.

Table 15—Basal rate schedule setting numeric object attributes

Attribute name	Extended configuration	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	{MDC_PART_PHD_DM | MDC_INS_BASAL_RATE_SCHED}	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	NR
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-agent-initiated | mss-cat-setting	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR
Unit-Code	MDC_DIM_INTL_UNIT_PER_HR.	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See ISO/IEEE 11073-20601:2016.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	O
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	M
Compound-Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	NR
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The value reported in this object is the basal rate schedule setting that represents a basal rate for a specifc schedule setting. The basal rate schedule setting shall be made available only through schedule objects.

The unit of a basal rate schedule setting is international units per hour.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as information on attribute ID and attribute type.

6.7.8 I:CHO schedule setting

Table 16 defines the attributes for the object that represents the I:CHO schedule values, which include the amount of carbohydrates in grams covered or disposed of by a single unit of insulin. The I:CHO schedule setting may be supported by the agent with extended configuration.

Table 16—I:CHO schedule setting attributes

Attribute name	Extended configuration	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	{MDC_PART_PHD_DM | MDC_INS_I2CHO_SCHED}	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	NR
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-agent-initiated | mss-cat-setting	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR
Unit-Code	MDC_DIM_G. See following text.	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See ISO/IEEE 11073-20601:2016.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	O
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	M
Compound-Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	NR
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The value reported in this object is the I:CHO for a specifc schedule setting. The I:CHO schedule setting shall be made available only through Schedule objects.

The unit of a I:CHO schedule setting is grams.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as information on attribute ID and attribute type.

6.7.9 ISF schedule setting

Table 17 defines the attributes for the object that represents the ISF schedule values. The ISF schedule setting may be supported by the agent with extended configuration.

Table 17—ISF schedule setting attributes

Attribute name	Extended configuration	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	{MDC_PART_PHD_DM | MDC_INS_ISF_SCHED}	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	NR
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-agent-initiated | mss-cat-setting	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR
Unit-Code	MDC_DIM_MILLI_MOLE_PER_L or MDC_DIM_MILLI_G_PER_DL. See following text.	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016	C
Source-Handle-Reference	See ISO/IEEE 11073-20601:2016.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	O
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	M
Compound-Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	NR
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The observed value reported in this object is ISF for a specifc schedule setting. The ISF schedule setting shall be made available only through schedule objects.

The unit of ISF schedule setting is millimole per liter or milligrams per deciliter.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as information on attribute ID and attribute type.

6.7.10 Insulin reservoir remaining

Table 18 defines the attributes for the object that represents the insulin remaining in the reservoir. The insulin reservoir remaining object may be supported by an insulin pump agent with extended configuration.

Table 18—Insulin reservoir remaining numeric object attributes

Attribute name	Extended configuration	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	{MDC_PART_PHD_DM | MDC_INS_RESERVOIR}	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	O
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-manager-initiated | mss-acc-agent-initiated | mss-cat-calculation	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See ISO/IEEE 11073-20601:2016.	O
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	C
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	O
Unit-Code	MDC_DIM_INTL_UNIT	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See ISO/IEEE 11073-20601:2016.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	O
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	R
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The observed value reported in this object is the amount of the insulin remaining in the reservoir. The insulin pump is responsible for determining the frequency of the reservoir remaining event to be reported (e.g., the insulin reservoir remaining may be reported at the same time as the bolus delivered or basal delivered measurements). Only non-negative numbers shall be used.

The insulin reservoir remaining numeric object does not support any methods, events, or other services.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as information on attribute ID and attribute type.

6.7.11 Insulin concentration

Table 19 defines the attributes for the object that represents the insulin concentration. The insulin concentration object may be supported by an insulin pump agent with extended configuration.

Table 19—Insulin concentration numeric object attributes

Attribute name	Extended configuration	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	{MDC_PART_PHD_DM | MDC_INS_CONC}	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	NR
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-acc-manager-initatied | mss-acc-agent-initiated | mss-cat-setting	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See ISO/IEEE 11073-20601:2016.	O
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	C
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	O
Unit-Code	MDC_DIM_INTL_UNIT_PER_ML, or MDC_DIM_INTL_UNIT_PER_L, or MDC_DIM_INTL_UNIT_PER_M_CUBE or MDC_DIM_INTL_UNIT_PER_CM_CUBE.	M
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See ISO/IEEE 11073-20601:2016.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	O
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Simple-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	R
Compound-Basic-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Compound-Nu-Observed-Value	See ISO/IEEE 11073-20601:2016.	C
Accuracy	See ISO/IEEE 11073-20601:2016.	NR

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The observed value reported in this object is the insulin concentration. Only non-negative numbers shall be used.

The insulin concentration numeric object does not support any methods, events, or other services.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as information on attribute ID and attribute type.

6.8 Real-time sample array objects

Real-time sample array objects are not required by this standard.

6.9 Enumeration objects

6.9.1 General

The enumeration objects listed in this subclause represent the enumeration observations that can be produced by an insulin pump agent. The nomenclature code to identify an enumeration object class is MDC_MOC_VMO_METRIC_ENUM. Subclauses 6.9.2, 6.9.3, and 6.9.4 define the precise definitions for operational status, general device status, and specific insulin pump status enumeration objects.

Enumeration objects do not support any methods, events, or other services.

6.9.2 Operational status

Table 20 defines the attributes for the object that represents the operational status. The operational status enumeration object may be supported by an insulin pump agent with extended configuration.

Table 20—Operational status enumeration object attributes

Attribute name	Extended configuration	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	MDC_PART_PHD_DM | MDC_INS_PUMP_OP_STAT	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	NR
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-acc-manager-initiated | mss-acc-agent-initiated	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR
Unit-Code	See ISO/IEEE 11073-20601:2016.	NR
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See following text.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	NR
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	NR
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value-Simple-OID	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value-Simple-Bit-Str	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value-Basic-Bit-Str	See following text.	R
Enum-Observed-Value-Simple-Str	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value-Partition	See ISO/IEEE 11073-20601:2016.	NR
Capability-Mask-Basic	See 6.2.	M
State-Flag-Basic	See 6.2.	M

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The observed value reported in this object is the operational status.

Because these are essentially flags, the Unit-Code attribute is not appropriate for this object. Similarly, Source-Handle-Reference is inappropriate, as this object monitors the status of the equipment.

The explicit expression of the existence of annunciations is realized by the setting of the appropriate bit in the Enum-Observed-Value-Basic-Bit-Str attribute, as defined in Table 21. If a manager supports the interpretation of this object, it shall be able to interpret the entire set of presented conditions. An agent shall report each time there is a change in status in a condition. An agent may implement any subset of these same conditions. Note that a manager shall interpret these bits only within the context of this attribute and only within this device specialization as other specializations may use corresponding terms for different purposes.

Table 21 defines the nomenclature terms for the operational and therapy conditions. If an acceptable, existing nomenclature term is not available, insulin-device-op-undetermined shall be used.

Table 21—Mapping of operational and therapy conditions to object Bit-Str attribute

Operational and therapy condition	OpStat mnemonic
Agent is in an undetermined operating condition.	insulin-device-op-undetermined
Agent is powering off (e.g., when the user has requested the insulin pump to turn off). No functionality available (e.g., no insulin delivery and no configuration of the device is possible).	insulin-device-op-off
Agent is in standby and most functionality can resume faster than from off condition (e.g., the device is in a condition to save power).	insulin-device-op-standby
Agent is performing initialization and preparing for infusion therapy (e.g., the insulin delivery device rewinds the piston rod to enable the insertion of a new reservoir or is detecting the position of the plunger in the reservoir).	insulin-device-op-preparing
Agent is preparing the delivery mechanism (e.g., filling the infusion set fluidic path with insulin after replacement of the reservoir and/or infusion set).	insulin-device-op-priming
Agent is ready to deliver therapy but waiting for an interaction (e.g., waiting for the infusion set to be connected to the body after priming or waiting for a user confirmation).	insulin-device-op-waiting
Agent is ready to administer insulin infusion therapy. The current therapy condition is further indicated by bits 9-11.	insulin-device-op-ready
Agent has stopped therapy and insulin delivery (e.g., bolus nor basal insulin is being administered), and current bolus setting is canceled.	insulin-device-therapy-stop
Agent has paused therapy momentarily and insulin delivery suspended (e.g., during preparing operation condition, a therapy pause can be used to bridge the time which is typically limited to several minutes during a reservoir change. When the agent returns to therapy run the remaining insulin amount is delivered).	insulin-device-therapy-pause
Agent is administerating and delivering insulin to patient. The current operation condition shall indicate “insulin-device-op-ready” (bit 6).	insulin-device-therapy-run

NOTE 1—The bits in the table above are defined to: 0 = False and 1 = True.

NOTE 2—The specific bit mappings of OpStat are defined in Annex B.

NOTE 3—All bits not defined in Table 21 or Annex B are reserved for future use.

NOTE 4—An agent is not required to implement all the features specified in Table 21.

At least one current therapy condition bit (insulin-device-therapy-stop, insulin-device-therapy-pause, or insulin-device-therapy-run) shall be set for any of the operating conditions, but the insulin-device-therapy-run condition shall only be set when the insulin-device-op-ready condition is set.

See ISO/IEEE 11073-20601:2016 for descriptive explanations on the individual attributes as well as information on attribute ID and attribute type.

6.9.3 PHD DM status

The PHD DM status object allows generic device events to be recorded in order to track important events for the user and troubleshooting information for manufacturers. Table 22 summarizes the attributes of the PHD DM status enumeration object. The PHD DM status enumeration object may be supported by an insulin pump agent.

Table 22—PHD DM status enumeration object attributes

Attribute name	Extended configuration	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	MDC_PART_PHD_DM | MDC_PHD_DM_DEV_STAT	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	Optional
Metric-Spec-Small	mss-avail-intermittent, mss-avail-stored-data, mss-upd-aperiodic, mss-acc-manager-initiated, mss-acc-agent-initiated.	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR
Unit-Code	See ISO/IEEE 11073-20601:2016.	NR
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See Table 23.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	NR
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	NR
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value-Simple-OID	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value-Simple-Bit-Str	See Table 23.	M
Capability-Mask-Simple	See 6.2.	M
State-Flag-Simple	See 6.2.	M

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The observed value reported in this object is the general device status.

Since these are essentially event flags, the Unit-Code attribute is not appropriate for this object. Similarly, Source-Handle-Reference is inappropriate, as this object monitors the status of the equipment.

The explicit expression of the existence of annunciations is realized by the setting of the appropriate bit in the Enum-Observed-Value-Simple-Bit-Str attribute, as defined in Table 23. If a manager supports the interpretation of this object, it shall be able to interpret the entire set of presented conditions. Anytime the status changes for any monitored condition, the agent shall report on all monitored conditions.

The detection of the condition change could take some time. As time stamp shall be used the correct start and stop time without the delay of the detection duration.

If an acceptable, existing bit is not available, device-status-undetermined shall be. A manager shall interpret these bits only within the context of this attribute and only within this device specialization, as other specializations may use corresponding terms for different purposes.

Table 23—Mapping of PHD DM status to object Bit-Str attribute

PHD DM status condition	PHDDMStat mnemonic
Agent reports that an undetermined or not supported condition occurred.	device-status-undetermined
Agent reports that a reset has occurred.	device-status-reset
Agent reports that a general fault occurred.	device-status-error
Agent reports that a mechanical fault occurred.	device-status-error-mechanical
Agent reports that an electronic fault occurred.	device-status-error-electronic
Agent reports that a software error occurred.	device-status-error-software
Agent reports that a battery fault occurred.	device-status-error-battery
Agent reports that a general service is required.	device-status-service
Agent reports that a time synchronization is required.	device-status-service-time-sync-required
Agent reports that a calibration is required.	device-status-service-calibration-required
Agent reports that a component replenishment is required.	device-status-service-replenishment-required
Agent reports that battery power is low.	device-status-battery-low
Agent reports that battery is depleted.	device-status-battery-depleted
Agent reports that battery has been replaced.	device-status-battery-replaced
Agent reports that battery is interrupted.	device-status-battery-interrupted

NOTE 1—The bits in the table above are defined to: 0 = False and 1 = True.

NOTE 2—The specific bit mappings of PHDDMStat are defined in Annex B.

NOTE 3—All bits not defined in Table 23 or Annex B are reserved for future use.

NOTE 4—An agent is not required to implement all the features specified in Table 23.

6.9.4 Insulin pump status

The insulin pump status object allows insulin pump specific information to be recorded in order to track important events for the user and troubleshooting information for manufacturers. Insulin pump notifications typically indicate conditions that inhibit proper insulin pump operation until the cause of the condition is eliminated.

Table 24 defines the attributes for the object that represents the insulin pump status. The insulin pump status enumeration object may be supported by an insulin pump agent in extended configurations.

If this object is to be implemented, then the nomenclature code to identify the enumeration object class is MDC_MOC_VMO_METRIC_ENUM. Refer to Table 24 for the set of attributes of this object.

Table 24—Insulin pump status enumeration object attributes

Attribute name	Extended configuration	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
Type	MDC_PART_PHD_DM | MDC_INS_PUMP_DEV_STAT	M
Supplemental-Types	See ISO/IEEE 11073-20601:2016.	NR
Metric-Spec-Small	mss-avail-intermittent | mss-avail-stored-data | mss-upd-aperiodic | mss-msmt-aperiodic | mss-acc-agent-initiated	M
Metric-Structure-Small	See ISO/IEEE 11073-20601:2016.	NR
Measurement-Status	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-List	See ISO/IEEE 11073-20601:2016.	NR
Metric-Id-Partition	See ISO/IEEE 11073-20601:2016.	NR
Unit-Code	See following text.	NR
Attribute-Value-Map	See ISO/IEEE 11073-20601:2016.	C
Source-Handle-Reference	See following text.	NR
Label-String	See ISO/IEEE 11073-20601:2016.	O
Unit-LabelString	See ISO/IEEE 11073-20601:2016.	O
Absolute-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Base-Offset-Time-Stamp	See ISO/IEEE 11073-20601:2016.	R
Relative-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
HiRes-Time-Stamp	See ISO/IEEE 11073-20601:2016.	C
Measure-Active-Period	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value-Simple-OID	See ISO/IEEE 11073-20601:2016.	C
Enum-Observed-Value-Simple-Bit-Str	See following text.	M
Enum-Observed-Value-Basic-Bit-Str	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value-Simple-Str	See ISO/IEEE 11073-20601:2016.	NR
Enum-Observed-Value	See following text.	NR
Enum-Observed-Value-Partition	See ISO/IEEE 11073-20601:2016.	O
Capability-Mask-Simple	See 6.2.	M
State-Flag-Simple	See 6.2.	M

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The observed value reported in this object is the insulin pump status.

Because these are essentially flags, the Unit-Code attribute is not appropriate for this object. Similarly, Source-Handle-Reference is inappropriate, as this object monitors the status of the equipment.

The explicit expression of the existence of annunciations is realized by the setting of the appropriate bit in the Enum-Observed-Value-Simple-Bit-Str attribute, as defined in Table 25. The Enum-Observed-Value attribute should not be used, as it unnecessarily complicates the modeling of the object. If a manager supports the interpretation of this object, it shall be able to interpret the entire set of presented conditions, defined in the InsPumpStat entity. An agent shall report each time there is a change in status in a condition. An agent may implement any subset of these same conditions. Note that a manager shall interpret these bits only within the context of this attribute and only within this device specialization as other specializations may use corresponding terms for different purposes.

Table 25 defines the nomenclature terms for the insulin pump status.

Table 25—Mapping of insulin pump status to object Bit-Str attribute

Insulin Pump condition	InsPumpStat mnemonic
Agent reports the air pressure is outside normal operating range.	air-pressure-out-of-range
Agent reports that active bolus canceled.	bolus-canceled
Agent detects delivery has reached a user-defined high threshold for delivery (basal, bolus, total).	delivery-max
Agent reports that infusion set (including tubing and/or cannula) is not attached to the body.	infusion-set-detached
Agent reports that the physical connection between infusion set (including tubing and/or cannula) and the insulin delivery device is incomplete.	infusion-set-incomplete
Agent reports an occlusion occurred and clogging of infusion set.	occlusion-detected
Agent reports an insufficient power to charge the device.	power-insufficient
Agent reports a priming issue (e.g., after replacement of reservoir and/or infusion set).	priming-issue
Agent reports that the reservoir is extremely low or empty and cannot continue insulin delivery.	reservoir-empty
Agent reports an error related to the replacement or functioning of the reservoir occurred (e.g., exceeded time out of the device, not properly attached or has been dislodged) and cannot continue insulin delivery.	reservoir-issue
Agent reports that the reservoir fill level reached a manufacturer defined low threshold.	reservoir-low
Agent reports that the reservoir was attached.	reservoir-attached
Agent reports that temporary basal rate canceled (e.g., device changed from operational to standby mode)	temp-basal-canceled
Agent reports that temporary basal rate expired.	temp-basal-expired
Agent reports the temperature is outside normal operating range.	temperature-out-of-range

NOTE 1—The bits in the table above are defined to: 0 = False and 1 = True.

NOTE 2—The specific bit mappings of InsPumpStat are defined in Annex B.

NOTE 3—All bits not defined in Table 25 or Annex B are reserved for future use.

NOTE 4—An agent is not required to implement all the features specified in Table 25.

6.10 PM-store objects

6.10.1 General

In the context of PHDs, insulin pump agent devices are portable or mobile devices. As stated in 5.1, insulin pump agents may be used to collect measurements or observations at a time when out of the network and agent/manager associations cannot be established. It is also common that a given set of measurements or observations made by insulin pump agents may need to be uploaded to more than one manager, for example, in the home and at a medical facility.

To support dual usage, an agent may provide two or more configurations. One configuration may use a temporary measurement storage model that uploads the most recent data immediately on association (agent initiated) with little user intervention, such as might be used by a typical home user that uploads measurements frequently to a personal computer or a mobile device such as a cell phone. Another configuration may use a long-term measurement storage model that uploads data at the request of the manager, such as might be used by the patient's physician or other HCP.

The long-term storage model is realized using PM-stores. Any configuration that does not include a PM-store object utilizes agent-initiated event reports to transmit the observations. The use of temporarily stored

data as defined in ISO/IEEE 11073-20601:2016 is most useful for small numbers of measurements and is subject to automatic deletion upon upload.

Alternatively, in the case where a large number of measurements may be stored or if automatic deletion is to be avoided, a PM-store configuration should be used. Any configuration with a PM-store for persistent storage shall enable access to the PM-store transmissions. As a result, this standard describes a mechanism using PM-store to hold measurements for longer durations. The data held in PM-store objects may be deleted by user actions via the manager or user interface on the device, and the capacity is limited only by the agent's data storage capabilities.

6.10.2 Persistent store model

The PM-store model defined by this standard utilizes one optional PM-store object, the metric PM-store object for the persistent storage of metric data (see Figure 5).

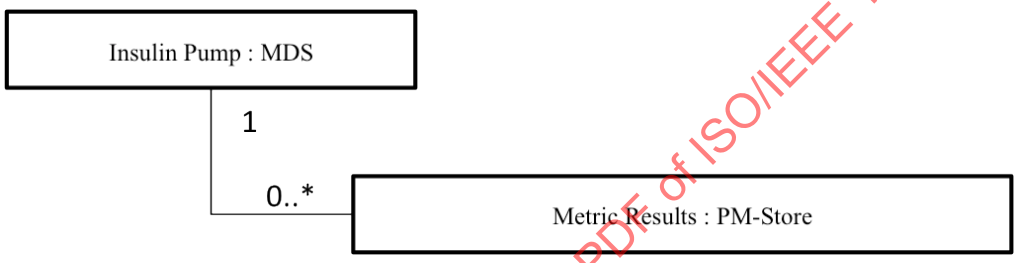

Figure 5—Insulin pump—persistent store model

Note that the PM-store objects are not part of standard configurations defined in this standard.

Following the guides provided in this standard should enable an implementer to store and retrieve the data within this model, but the specifics for determining the specific nature of the data layout and the subsequent visualization, mining, or other managing of the retrieved data is outside the scope of this standard.

The PM-store model defined by this standard utilizes one or more PM-segments for the data of each object to be persistently stored (see Figure 6, for example).

For an insulin pump, the metric PM-store model (see Figure 6) consists of segments holding metric and enumeration readings. The segments holding readings from the objects for bolus delivered and current basal rate setting shall be supported if the metric PM-store is implemented. The other segments are optional and hold observations from the objects that are instantiated.

Note that the metric PM-store object is not part of standard configurations defined in this standard.

Each entry in a segment shall include one of the time formats in the segm-entry-header so a manager can correlate entries across the different segments. If a particular object is not supported, the corresponding segment is not required to exist. Each segment has a cardinality of zero-to-many or one-to-many, as PM-segments are required to contain data from a contiguous period of time (see ISO/IEEE 11073-20601:2016). Therefore changing time and/or date on the agent typically results in the creation of new segment instances for the supported measurement or observation objects. Furthermore, an insulin pump agent may subdivide data from one contiguous period of time into several segments for further clustering of data (e.g., one segment per day or for an uninterrupted time span of the insulin pump being in operating mode). If a particular segment resulting from such time/date changes or clustering does not contain any entries, it is not required to exist.

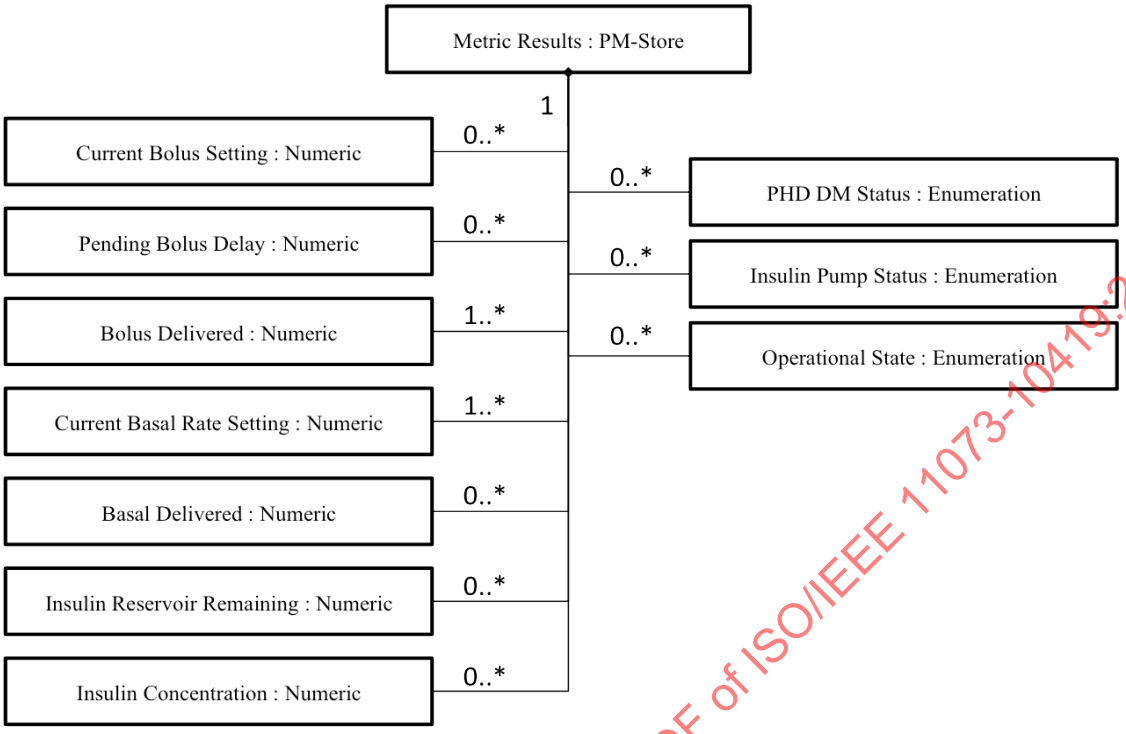

Figure 6—Insulin pump—metric persistent store model

6.10.3 Metric results PM-store object attributes

Table 26 summarizes the attributes of the metric results PM-store object. The nomenclature code to identify the PM-store object is MDC_MOC_VMO_PMSTORE.

Table 26—Metric results PM-store object attributes

Attribute name	Extended configuration	
	Value	Qualifier
Handle	See ISO/IEEE 11073-20601:2016.	M
PM-Store-Capab	See ISO/IEEE 11073-20601:2016.	M
Store-Sample-Algorithm	See ISO/IEEE 11073-20601:2016.	M
Store-Capacity-Count	See ISO/IEEE 11073-20601:2016.	M
Store-Usage-Count	See ISO/IEEE 11073-20601:2016.	M
Operational-State	See ISO/IEEE 11073-20601:2016.	M
PM-Store-Label	See ISO/IEEE 11073-20601:2016.	O
Sample-Period	See ISO/IEEE 11073-20601:2016.	NR
Number-Of-Segments	See ISO/IEEE 11073-20601:2016.	M
Clear-Timeout	See ISO/IEEE 11073-20601:2016.	M

The PM-Store-Capab attribute shall set the following bits as indicated:

— **pmsc-var-no-of-segm:**

If the agent creates new segments either due to storing data of multiple sessions or due to time changes as described in 8.12.2.2 of ISO/IEEE 11073-20601:2016, then pmsc-var-no-of-segm shall be set.

— **pmsc-epi-seg-entries:**

The pmsc-epi-seg-entries bit shall be set.

— **pmsc-peri-seg-entries:**

The pmsc-peri-seg-entries bit shall not be set.

The remaining bits of the PM-Store-Capab attribute are agent specific and shall be set appropriately.

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

6.10.4 PM-store object methods

Table 27 defines the methods of the PM-store objects.

Table 27—PM-store object methods

Service	Subservice type name	Mode	Subservice type (action-type)	Parameters (action-info-args)	Results (action-info-args)
ACTION	Clear-Segments	Confirmed	MDC_ACT_SEG_CLR	SegmSelection	
	Get-Segment-Info	Confirmed	MDC_ACT_SEG_GET_INFO	SegmSelection	SegmentInfoList
	Get-Segment-Id-List	Confirmed	MDC_ACT_SEG_GET_ID_LIST	(empty)	SegmIdList
	Trig-Segment-Data-Xfer	Confirmed	MDC_ACT_SEG_TRIG_XFER	TrigSegmDataXferReq	TrigSegmDataXferRsp

6.10.4.1 Clear-Segments

This method allows the manager to delete all data entries stored in a PM-segment object. The agent shall support the Clear-Segments method by setting the pmsc-clear-segm-by-all-sup bit for the PM-Store-Capab attribute. Deletion of PM-segments is not guaranteed by this method. See ISO/IEEE 11073-20601:2016 for information on how the agent shall reply in case it decides to protect certain segments from deletion.

— **Get-Segment-Info:**

This method allows the manager to retrieve the PM-segment attributes.

— **Get-Segment-Id-List:**

This method allows the manager to retrieve a list of the instance numbers of all the PM-segments of a PM-store.

— **Trig-Segment-Data-Xfer:**

This method allows the manager to initiate the transfer of the data entries stored in the PM-segment object. Refer to ISO/IEEE 11073-20601:2016 for details.

6.10.5 PM-store object events

Table 28 defines the events sent by the PM-store objects.

Table 28—PM-store object events

Service	Subservice type name	Mode	Subservice type (event-type)	Parameters (event-info)	Results (event-reply-info)
EVENT REPORT	Segment-Data-Event	Confirmed	MDC_NOTI_SEGMENT_DATA	SegmentDataEvent	SegmentDataResult

— **Segment-Data-Event:**

This event allows the agent to send the data entries stored in the PM-segment object. This event is triggered by the manager using the Trig-Segment-Data-Xfer action. Refer to ISO/IEEE 11073-20601:2016 for details.

6.10.6 PM-store object services

6.10.6.1 GET service

The GET service shall be provided by an agent implementing PM-store objects. This service shall be available only while the agent is in the Operating state. Refer to ISO/IEEE 11073-20601:2016 for details.

6.10.6.2 SET service

There are currently no SET services defined for PM-store objects in this standard.

6.10.7 PM-segment objects

NOTE 1— defines the attributes of the periodic session PM-segment object contained in the periodic PM-store object managing the stored measurements or observations. The nomenclature code to identify the PM-segment class is MDC_MOC_PM_SEGMENT.

NOTE 1—See ISO/IEEE 11073-20601:2016 for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The Fixed-Segment-Data attribute serves as the container of the stored measurements or observations. When the Fixed-Segment-Data attribute is transmitted, all entries in the event report are formatted according to the PM-Segment-Entry-Map. Each entry contains an optional header and one or more elements. Each element holds data from one or more metric measurements.

Table 29—Common PM-segment object attributes

Attribute name	Extended configuration	
	Value	Qualifier
Instance-Number	See ISO/IEEE 11073-20601:2016.	M
PM-Segment-Entry-Map	See ISO/IEEE 11073-20601:2016.	M
PM-Seg-Person-Id	See ISO/IEEE 11073-20601:2016.	C
Operational-State	See ISO/IEEE 11073-20601:2016.	M
Sample-Period	See ISO/IEEE 11073-20601:2016.	C
Segment-Label	See ISO/IEEE 11073-20601:2016.	O
Segment-Start-Abs-Time	See ISO/IEEE 11073-20601:2016.	C
Segment-End-Abs-Time	See ISO/IEEE 11073-20601:2016.	C
Date-and-Time-Adjustment	See ISO/IEEE 11073-20601:2016.	C
Segment-Start-BO-Time	See ISO/IEEE 11073-20601:2016.	C
Segment-End-BO-Time	See ISO/IEEE 11073-20601:2016.	C
Segment-Usage-Count	See ISO/IEEE 11073-20601:2016.	M
Segment-Statistics	See ISO/IEEE 11073-20601:2016.	O
Fixed-Segment-Data	Segment data transferred as an array of entries in a format as specified in the PM-Segment-Entry-Map attribute.	M
Confirm-Timeout	See ISO/IEEE 11073-20601:2016.	O
Transfer-Timeout	See ISO/IEEE 11073-20601:2016.	M

6.11 Schedule-store objects

6.11.1 General

In the context of PHDs, insulin pump agent devices can store schedules at a time when agent/manager are associated or when out of the network and agent/manager associations cannot be established. It is also common that a given set of schedules made by insulin pump agents could be uploaded to more than one manager, for example, in the home and at a medical facility.

The data held in schedule-store objects may be deleted by user actions via the manager or user interface on the device, and the capacity is limited only by the agent's data storage capabilities.

6.11.2 Schedule-store model

The schedule-store model defined by this standard utilizes three optional schedule-store objects:

— Basal profile settings schedule-store

— I:CHO profile settings schedule-store

— ISF profile settings schedule-store for the storage of basal profile settings, I:CHO profile settings, and ISF profile settings, respectively (see Figure 7)

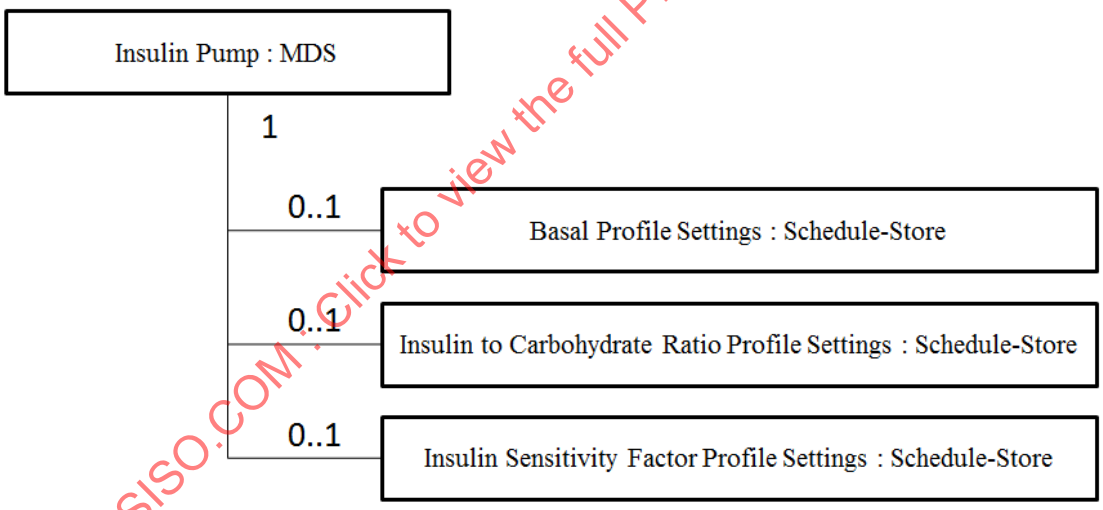

Figure 7—Insulin pump—schedule-store model

Note that the schedule-store objects are not part of standard configurations defined in this standard.

Following the guides provided in this standard should enable an implementer to store and retrieve the data within this model, but the specifics for determining the specific nature of the data layout and the subsequent visualization, mining, or other managing of the retrieved data is outside the scope of this standard.

6.11.3 Basal profile settings schedule-store object attributes

For an insulin pump, the basal profile settings schedule-store model (see Figure 8) may consist of zero or more basal profile sched schedule-segments to account for the collection of multiple basal profile settings.

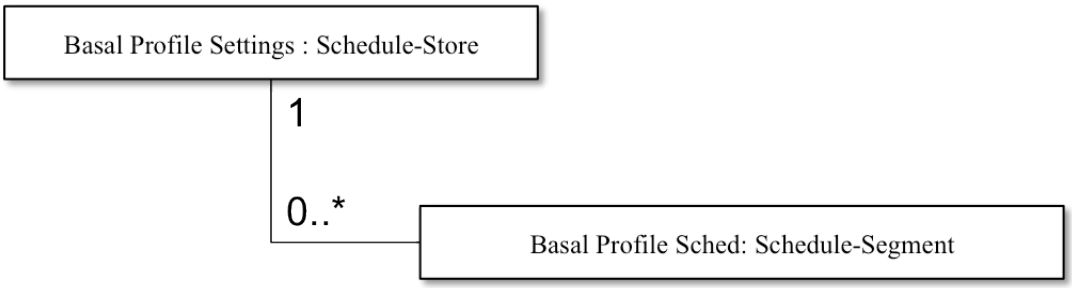

Figure 8—Insulin pump—basal profile settings schedule-store model

If the insulin pump device offers the option of programming multiple basal profiles, a unique basal profile identifier is captured in the basal profile sched schedule-segment and can be used to identify the different collections of basal profile settings. The active basal profile sched is captured in the basal profile settings schedule-store.

Within a basal profile sched schedule-segment, at least one entry shall be used to account for a basal rate schedule setting.

Figure 9 is an example of a basal rate profile setting schedule-store.

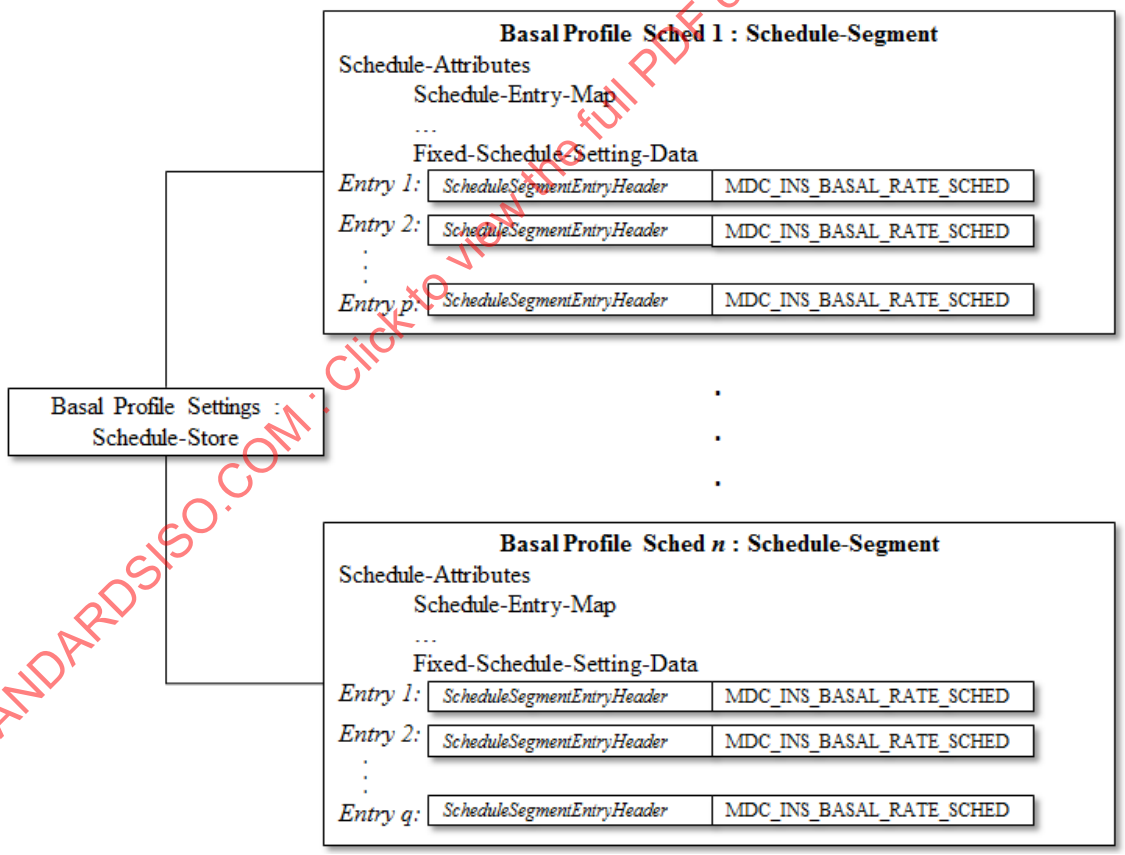

Figure 9—Basal profile settings schedule-store

Note that the basal profile settings schedule-store object is not part of standard configurations defined in this standard.

Table 30 summarizes the attributes of the basal profile settings schedule-store object. The nomenclature code to identify the basal profile settings schedule-store object is MDC_MOC_VMO_SCHEDSTORE.

Table 30—Basal profile settings schedule-store object attributes

Attribute name	Extended configuration	
	Value	Qualifier
Schedule-Handle	See Annex E.	M
Active-Schedule-Segment-Instance-Number	See Annex E.	M
Updated-Schedule-Segment-Instance-Number-List	See Annex E.	M
Schedule-Store-Capab	See Annex E.	M
Schedule-Store-Capacity-Count	See Annex E.	M
Schedule-Store-Usage-Count	See Annex E.	M
Schedule-Store-Operational-Status	See Annex E.	M
Schedule-Store-Label	See Annex E.	O
Number-Of-Schedule-Segments	See Annex E.	M

The basal profile settings Schedule-Store-Capab attribute shall set the following bits as indicated:

— **schedsc-var-no-of-segm:**

If the agent creates new schedule-segments either due to storing data of multiple schedules or due to time changes as described in 8.12.2.2 of ISO/IEEE 11073-20601:2016, then schedsc-var-no-of-segm shall be set.

— **schedsc-epi-seg-entries:**

The schedsc-epi-seg-entries bit shall be set.

— **schedsc-peri-seg-entries:**

The schedsc-peri-seg-entries bit shall not be set.

The remaining bits of the Schedule-Store-Capab attribute are agent specific and shall be set appropriately.

NOTE 1—See Annex E for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

6.11.4 I:CHO profile settings schedule-store object attributes

For an insulin pump, the I:CHO profile settings schedule-store model (see Figure 10) may consist of zero or more I:CHO profile sched schedule-segments to account for the collection of multiple I:CHO profile settings.

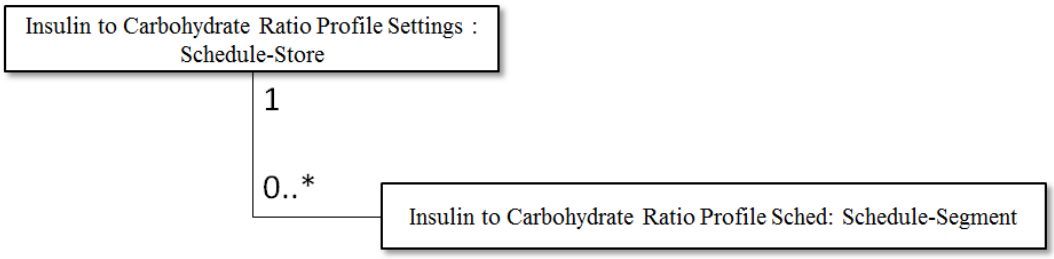

Figure 10—I:CHO profile settings schedule-store model

If the insulin pump device offers the option of programming multiple I:CHO profiles, the unique I:CHO profile identifier is captured in the I:CHO profile sched schedule-segment and can be used to identify the different I:CHO profile settings. The active I:CHO profile sched is captured in the I:CHO profile sched schedule-store.

Within an I:CHO profile sched schedule-segment, at least one entry shall be used to account for an I:CHO schedule setting.

Figure 11 is an example of an I:CHO profile setting schedule-store.

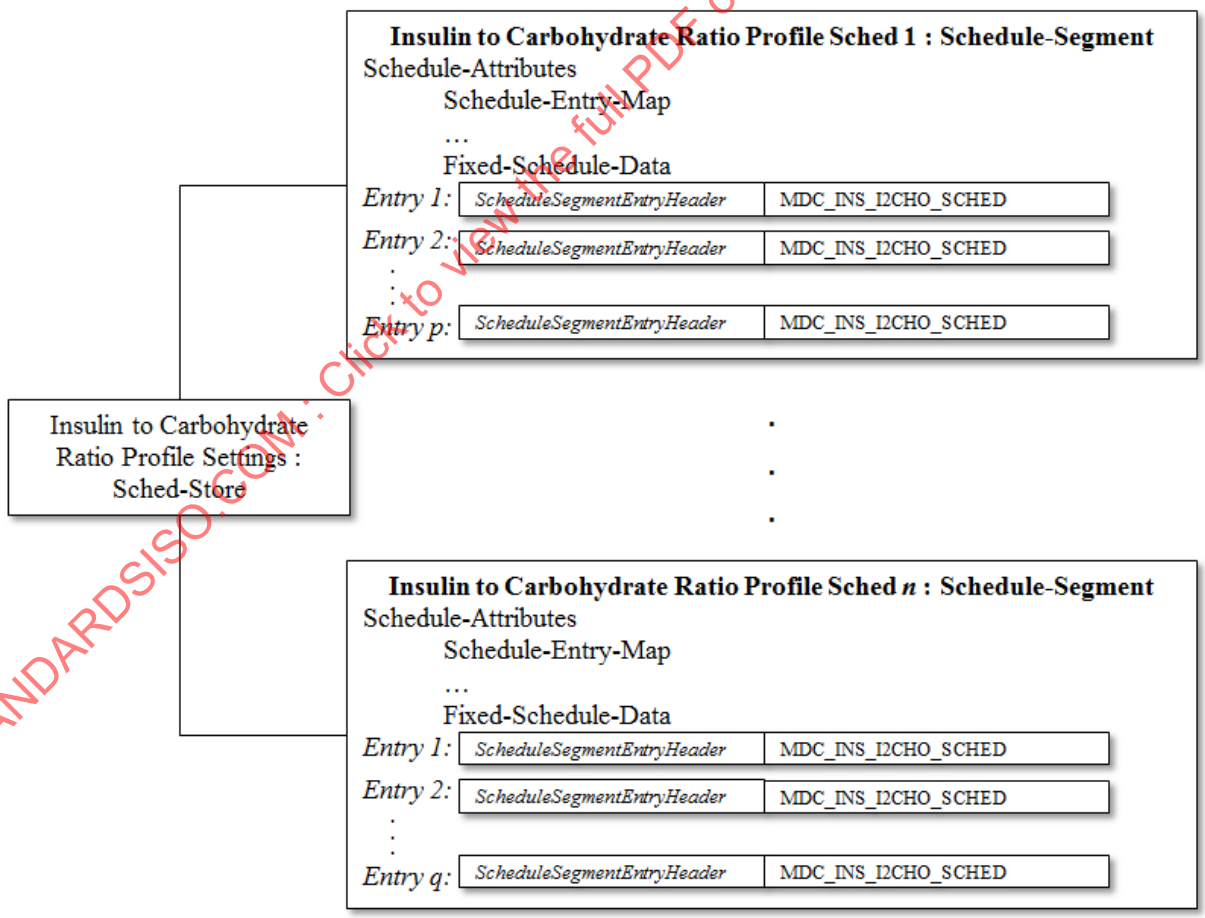

Figure 11—I:CHO profile settings schedule-store

Note that the I:CHO profile settings schedule-store object is not part of standard configurations defined in this standard.

Table 31 summarizes the attributes of the I:CHO profiles schedule-store object. The nomenclature code to identify the I:CHO profile settings schedule-store object is MDC_MOC_VMO_SCHEDSTORE.

Table 31—I:CHO profile settings schedule-store object attributes

Attribute name	Extended configuration	
	Value	Qualifier
Schedule-Handle	See Annex E.	M
Active-Schedule-Segment-Instance-Number	See Annex E.	M
Updated-Schedule-Segment-Instance-Number-List	See Annex E.	M
Schedule-Store-Capab	See Annex E.	M
Schedule-Store-Capacity-Count	See Annex E.	M
Schedule-Store-Usage-Count	See Annex E.	M
Schedule-Store-Operational-Status	See Annex E.	M
Schedule-Store-Label	See Annex E.	O
Number-Of-Schedule-Segments	See Annex E.	M

The I:CHO profile settings Schedule-Store-Capab attribute shall set the following bits as indicated:

— **schedsc-var-no-of-segm:**

If the agent creates new schedule-segments either due to storing data of multiple schedules or due to time changes as described in 8.12.2.2 of ISO/IEEE 11073-20601:2016, then schedsc-var-no-of-segm shall be set.

— **schedsc-epi-seg-entries:**

The schedsc-epi-seg-entries bit shall be set.

— **schedsc-peri-seg-entries:**

The schedsc-peri-seg-entries bit shall not be set.

The remaining bits of the Schedule-Store-Capab attribute are agent specific and shall be set appropriately.

NOTE 1—See Annex E for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

6.11.5 ISF profiles schedule-store object attributes

For an insulin pump, the ISF profile settings schedule-store model (see Figure 12) may consist of zero or more ISF profile sched schedule-segments to account for the data collection of multiple ISF profile settings.

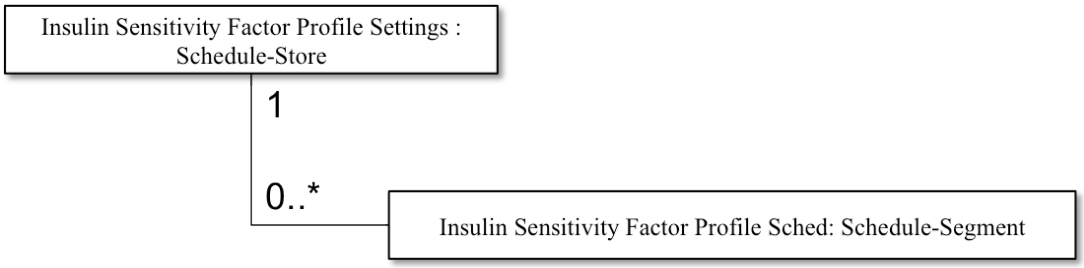

Figure 12—ISF profile settings schedule-store model

If the insulin pump device offers the option of programming multiple ISF profiles, the unique ISF profile identifier is captured in the ISF profile sched schedule-segment and can be used to identify the different ISF profile settings. The active ISF profile sched is captured in the ISF profile sched schedule-store.

Within an ISF profile sched schedule-segment, at least one entry shall be used to account for an ISF schedule setting.

Figure 13 is an example of an ISF profile setting schedule-store.

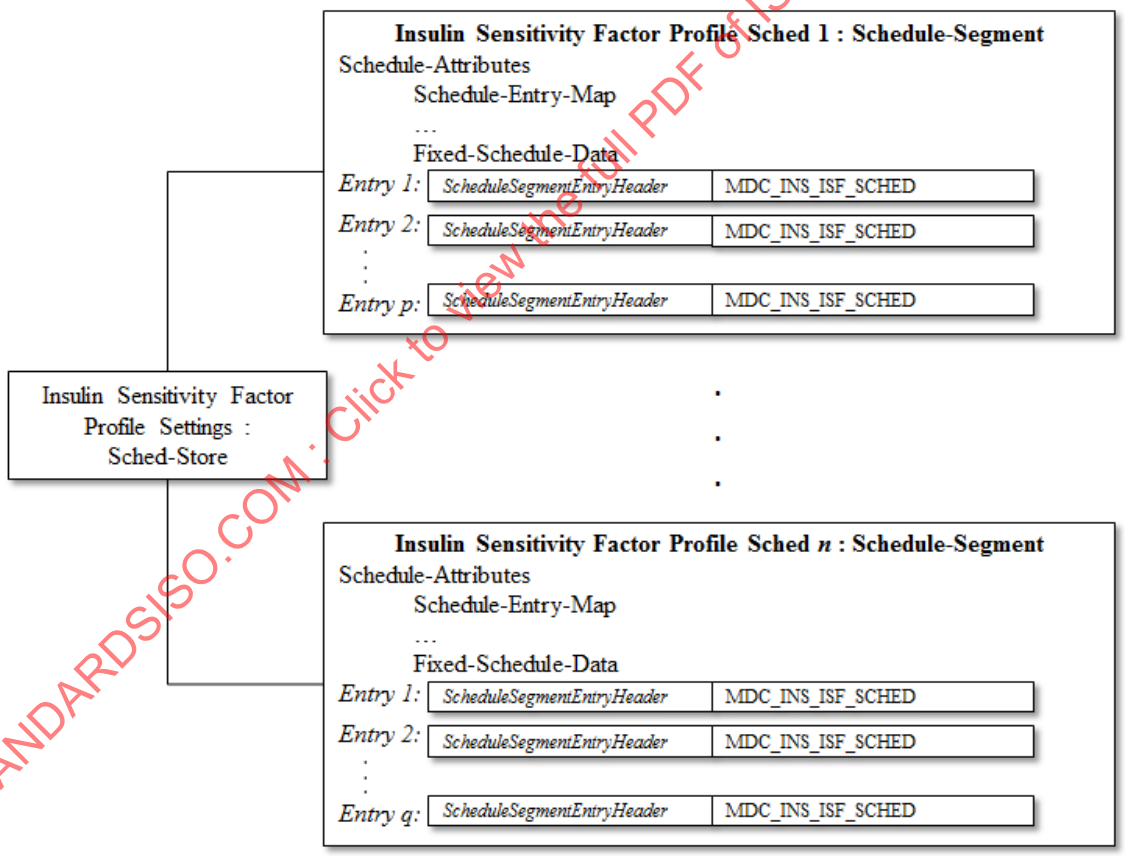

Figure 13—ISF profile settings schedule-store

Note that the ISF profile settings schedule-store object is not part of standard configurations defined in this standard.

Table 32 summarizes the attributes of the ISF profile settings schedule-store object. The nomenclature code to identify the ISF profile settings schedule-store object is MDC_MOC_VMO_SCHEDSTORE.

Table 32—ISF profile settings schedule-store object attributes

Attribute name	Extended configuration	
	Value	Qualifiers
Schedule-Handle	See Annex E.	M
Active-Schedule-Segment-Instance-Number	See Annex E.	M
Updated-Schedule-Segment-Instance-Number-List	See Annex E.	M
Schedule-Store-Capab	See Annex E.	M
Schedule-Store-Capacity-Count	See Annex E.	M
Schedule-Store-Usage-Count	See Annex E.	M
Schedule-Store-Operational-Status	See Annex E.	M
Schedule-Store-Label	See Annex E.	O
Number-Of-Schedule-Segments	See Annex E.	M

The ISF profile settings Schedule-Store-Capab attribute shall set the following bits as indicated:

— **schedsc-var-no-of-segm:**

If the agent creates new schedule-segments either due to storing data of multiple schedules or due to time changes as described in 8.12.2.2 of ISO/IEEE 11073-20601:2016, then schedsc-var-no-of-segm shall be set.

— **schedsc-epi-seg-entries:**

The schedsc-epi-seg-entries bit shall be set.

— **schedsc-peri-seg-entries:**

The schedsc-peri-seg-entries bit shall not be set.

The remaining bits of the Schedule-Store-Capab attribute are agent specific and shall be set appropriately.

NOTE 1—See Annex E for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

6.11.6 Schedule-store object methods

This subclause applies to basal profile settings, I:CHO profile settings, and ISF profile settings schedule-store objects.

Table 33 defines the methods of a schedule-store object.

Table 33—Common schedule-store object methods

Service	Subservice type name	Mode	Subservice type (action-type)	Parameters (action-info-args)	Results (action-info-args)
ACTION	Get-Schedule-Segment-Info	Confirmed	MDC_ACT_SCHED_SEG_GET_INFO	SchedSegmSelection	SchedSegmentInfoList
	Get-Schedule-Segment-Id-List	Confirmed	MDC_ACT_SCHED_SEG_GET_ID_LIST	(empty)	SchedSegmIdList
	Trig-Schedule-Segment-Data-Xfer	Confirmed	MDC_ACT_SCHED_SEG_TRIG_XFER	TrigSchedSegmDataXferReq	TrigSchedSegmDataXferRsp

— **Get-Schedule-Segment-Info:**

This method allows the manager to retrieve schedule-segment attributes.

— **Get-Schedule-Segment-Id-List:**

This method allows the manager to retrieve a list of the instance numbers of all the schedule-segments of a schedule-store.

— **Trig-Schedule-Segment-Data-Xfer:**

This method allows the manager to initiate the transfer of the data entries stored in a schedule-segment object.

Refer to Annex E for details.

6.11.7 Schedule-store object events

This subclause applies to basal profile settings, I:CHO profile settings, and ISF profile settings schedule-store objects.

Table 34 defines the events sent by schedule-store objects.

Table 34—Insulin pump schedule-store object events

Service	Subservice type name	Mode	Subservice type (event-type)	Parameters (event-info)	Results (event-reply-info)
EVENT REPORT	Schedule-Segment-Data-Event	Confirmed	MDC_NOTI_SCHED_SEGMENT_DATA	ScheduleSegmentDataEvent	ScheduleSegmentDataResult

— **Schedule-Segment-Data-Event:**

This event allows the agent to send the data entries stored in a schedule-segment object. This event is triggered by the manager using the Trig-Schedule-Segment-Data-Xfer action. Refer to E.1.5.

6.11.8 Schedule-store object services

This subclause applies to basal profile settings, I:CHO profile settings, and ISF profile settings schedule-store objects.

6.11.8.1 GET service

The GET service shall be provided by an agent implementing schedule-store objects. This service shall be available only while the agent is in the Operating state. Refer to E.1.4 for details.

6.11.8.2 SET service

There are currently no SET services defined for schedule-store objects in this standard.

6.11.9 Schedule-segment objects

This subclause applies to basal profile sched, I:CHO profile sched, and ISF profile sched schedule-segment objects.

Each of the schedule-store objects contains at least one or more corresponding schedule-segment object.

Table 35 defines the attributes schedule-segment object contained in the schedule-store object managing the stored schedules. The nomenclature code to identify the schedule segment class is MDC_MOC_SCHEDULE_SEGMENT.

Table 35—Schedule-segment object attributes

Attribute name	Extended configuration	
	Value	Qualifier
Schedule-Segment-Instance-Number	See Annex E.	M
Schedule-Segment-Entry-Map	See Annex E.	M
Schedule-Segment-Period	See Annex E.	M
Schedule-Segment-Entry-Interval	See Annex E.	O
Schedule-Segment-Person-Id	See Annex E.	O
Schedule-Segment-Entry-Count	See Annex E.	O
Schedule-Segment-Label	See Annex E.	O
Schedule-Segment-LastUpdated-Abs-Time	See Annex E.	C
Schedule-Segment-LastUpdated-HiRes-Time	See Annex E.	C
Schedule-Segment-LastUpdated-BO-Time	See Annex E.	R
Schedule-Segment-Reference-Abs-Time	See Annex E.	C
Schedule-Segment-Reference-BO-Time	See Annex E.	R
Schedule-Segment-Start-Abs-Time	See Annex E.	C
Schedule-Segment-End-Abs-Time	See Annex E.	C
Schedule-Segment-Start-BO-Time	See Annex E.	R
Schedule-Segment-End-BO-Time	See Annex E.	R
Fixed-Schedule-Segment-Data	Schedule-segment data transferred as an array of entries in a format as specified in the Schedule-Segment-Entry-Map attribute.	M
Schedule-Segment-Confirm-Timeout	See Annex E.	O
Schedule_Transfer-Timeout	See Annex E.	M

NOTE 1—See Annex E for information on whether an attribute is static or dynamic.

NOTE 2—See 6.3 for a description of the qualifiers.

The Fixed-Schedule-Segment-Data attribute serves as the container of the schedules. When the Fixed-Schedule-Segment-Data attribute is transmitted, all entries in the event report are formatted according to the Schedule-Segment-Entry-Map. Each entry stores a single sample point, which may consist of a set of attributes.

6.12 Scanner objects

Scanner objects are not required by this standard.

6.13 Class extension objects

In this standard, no class extension objects are defined with respect to ISO/IEEE 11073-20601:2016.

6.14 Insulin pump information model extensibility rules

The insulin pump DIM of this standard may be extended by including elements defined in ISO/IEEE 11073-20601:2016 as well as vendor-specific elements. Any object or attribute extensions implemented should follow the guidelines of this standard as closely as possible.

An insulin pump agent having a configuration with extensions beyond the standard configuration, as specified in this standard, shall use a configuration ID in the range of IDs reserved for extended configurations (see ISO/IEEE 11073-20601:2016).

7. Insulin pump service model

7.1 General

The service model defines the conceptual mechanisms for data exchange services. These services are mapped to messages that are exchanged between the agent and the manager. Protocol messages within the ISO/IEEE 11073 series of standards are defined in ASN.1. See ISO/IEEE 11073-20601:2016 for a detailed description of the PHD service model. Subclauses 7.2 and 7.3 define the specifics of object access and event reporting services for an insulin pump agent according to this standard.

7.2 Object access services

The object access services of ISO/IEEE 11073-20601:2016 are used to access the objects defined in the DIM of the insulin pump.

The following generic object access services are supported by an insulin pump agent according to this standard:

— GET service: used by the manager to retrieve the values of the agent MDS object or PM-store attributes or schedule-store attributes. The list of insulin pump attributes is given in 6.6.4.1 for the MDS object, and 6.10.3 for the PM-store objects, and 6.11.3, 6.11.4, and 6.11.5 for the schedule-store objects.

— SET service: used by the manager to set the values of the agent object attributes. There are no settable attributes defined for an insulin pump agent according to this standard.

— Event report service: used by the agent to send configuration reports and measurement data to the manager. The list of event reports for the insulin pump device specialization is given for the MDS object in 6.6.3, for the PM-store object in 6.10.5, and for the schedule-store object in 6.11.7.

— Action service: used by the manager to invoke actions (or methods) supported by the agent. The list of MDS objects actions for setting the time is given in 6.6.2. The list of PM-store object actions is given in 6.10.4. The list of schedule-store object actions is given in 6.11.6.

Table 36 summarizes the object access services described in this standard.

Table 36—Insulin pump object access services

Service	Subservice type name	Mode	Subservice type	Parameters	Result	Remarks
GET	<na>	<implied Confirmed>	<na>	GetArgumentSimple = (obj-handle = 0), attribute-id-list <*optional*>	GetResultSimple = (obj-handle = 0), attribute-list	Allows the manager to retrieve the value of attributes of the MDS object in the agent.
	<na>	<implied Confirmed>	<na>	GetArgumentSimple = (obj-handle = *handle of PM-store object*), attribute-id-list <*optional*>	GetResultSimple = (obj-handle = *handle of PM-store object*), attribute-list	Allows the manager to retrieve the values of attributes of a PM-store object in the agent.
	<na>	<implied Confirmed>	<na>	GetArgumentSimple = (obj-handle = *handle of Schedule-store object*), attribute-id-list <*optional*>	GetResultSimple = (obj-handle = *handle of Schedule-store object*), attribute-list	Allows the manager to retrieve the values of attributes of a Schedule-store object in the agent.
EVENT REPORT	MDS-Configuration-Event	Confirmed	MDC_NOTI_CONFIG	ConfigReport	ConfigReportRsp	Configuration Report to inform manager of the configuration of the agent.
	MDS-Dynamic-Data-Update-Var	Confirmed	MDC_NOTI_SCAN_REPORT_VAR	ScanReportInfoVar	—	Data Report to provide dynamic data to manager for some or all of the agent's objects in variable format.
	MDS-Dynamic-Data-Update-Fixed	Confirmed	MDC_NOTI_SCAN_REPORT_FIXED	ScanReportInfoFixed	—	Data Report to provide dynamic data to manager for some or all of the agent's objects in fixed format.
	MDS-Dynamic-Data-Update-MP-Var	Confirmed	MDC_NOTI_SCAN_REPORT_MP_VAR	ScanReportInfoMPVar	—	This is the same as MDS-Dynamic-Data-Update-Var but allows inclusion of data from multiple people.
	MDS-Dynamic-Data-Update-MP-Fixed	Confirmed	MDC_NOTI_SCAN_REPORT_MP_FIXED	ScanReportInfoMPFixed	—	This is the same as MDS-Dynamic-Data-Update-Fixed but allows inclusion of data from multiple people.
	Segment-Data-Event	Confirmed	MDC_NOTI_SEGMENT_DATA	SegmentDataEvent	SegmentDataResult	PM-store object event to provide data stored in the Fixed-Segment-Data of a PM-segment from the agent to the manager.
	Schedule-Segment-Data-Event	Confirmed	MDC_NOTISCHED_SEGMENT_DATA	ScheduleSegmentDataEvent	ScheduleSegmentDataResult	Schedule-store object event to provide data stored in the Fixed-Schedule-Segment-Data of a Schedule-segment from the agent to the manager.

(Table continues on the next page.)

Table 36—Insulin pump object access services *(continued)*

Service	Subservice type name	Mode	Subservice type	Parameters	Result	Remarks
ACTION	Set-Time	Confirmed	MDC_ACT_SET_TIME	SetTimeInvoke	—	Manager method to invoke the agent to set time in absolute time format to requested value.
	Set-Base-Offset-Time	Confirmed	MDC_ACT_SET_BO_TIME	SetBOTime Invoke	—	Manager method to invoke the agent to set time in base offset time format to requested value.
	Clear-Segments	Confirmed	MDC_ACT_SEG_CLR	SegmSelection	—	Allows the manager to delete data stored in selected PM-segments in the agent.
	Get-Segment-Info	Confirmed	MDC_ACT_SEG_GET_INFO	SegmSelection	SegmentInfoList	Allows the manager to retrieve the value of PM-segment attributes of one or more PM-segments in the agent.
	Get-Segment-Id-List	Confirmed	MDC_ACT_SEG_GET_ID_LIST	(empty)	SegmIdList	Allows the manager to retrieve a list of the instance numbers of all the PM-segments of a PM-store.
	Get-Schedule-Segment-Info	Confirmed	MDC_ACT_SCHED_SEG_GET_INFO	SehedSegmSelection	SchedSegmentInfoList	Allows the manager to retrieve the value of Schedule-segment attributes of one or more Schedule-segments in the agent.
	Get-Schedule-Segment-Id-List	Confirmed	MDC_ACTSCHED_SEG_GET_ID_LIST	(empty)	SchedSegmIdList	Allows the manager to retrieve a list of the instance numbers of all the Schedule-segments of a Schedule-store.
	Trig-Segment-Data-Xfer	Confirmed	MDC_ACT_SEG_TRIG_XFER	TrigSegmDataXferReq	TrigSegmDataXferRsp	Allows the manager to start the transfer of the Fixed-Segment-Data attribute of a PM-segment in the agent.
	Trig-Schedul-Segment-Data-Xfer	Confirmed	MDC_ACT_SCHED_SEG_TRIG_XFER	TrigSchedSegmDataXferReq	TrigSchedSegmDataXferRsp	Allows the manager to start the transfer of the Fixed-Schedule-Segment-Data attribute of a Schedule-segment in the agent.

7.3 Object access event report services

The event report service (see Table 36) is used by the agent to report its information (e.g., measurements). Event reports in this standard are a property of the MDS object only. The event reports used in this standard are defined in ISO/IEEE 11073-20601:2016.

The following conditions apply for an insulin pump agent according to this standard:

— MDS event reports shall be used in confirmed mode.

— Agent initiated mode shall be supported for measurement data transmission.

— Persistently stored metric mode may be supported for measurement data transmission.

— Manager initiated mode may be support for measurement data transmission.

An insulin pump agent, which is designed to operate in an environment where data may be collected from multiple people, may use one of the multiple-person event report styles to transmit all the data from each person in a single event. If this functionality is not required, the agent may use the single-person event report styles, which have reduced overhead.

A manager shall support both single-person and multiple-person event reports. An insulin pump agent may support either one or both single-person and multiple-person event reports. The formats for single-person and multiple-person reports are described in ISO/IEEE 11073-20601:2016.

8. Insulin pump communication model

8.1 Overview

This clause describes the general communication model and procedures of the insulin pump agent as defined in ISO/IEEE 11073-20601:2016. Therefore, the respective parts of ISO/IEEE 11073-20601:2016 are not reproduced; rather the specific choices and restrictions with respect to optional elements (e.g., objects, attributes, and actions) and specific extensions (e.g., nomenclature terms) are specified.

For an illustrative overview of the various message transactions during a typical measurement session, see the sequence diagram for the example use case in Annex D and the corresponding protocol data unit (PDU) examples in Annex I.

8.2 Communications characteristics

In this subclause, limits on the size of an application protocol data unit (APDU) transmitted or to be received by an insulin pump agent are defined. Small limits allow for simple implementations in terms of low cost and complexity.

An insulin pump agent implementing only this device specialization shall not transmit any APDU larger than N_{tx} and shall be capable of receiving any APDU up to a size of N_{rx}. For this standard, N_{tx} shall be 64 512 octets for implementations supporting persistent metric storage. In the absence of the persistent metric storage capability, an insulin pump agent implementing only this device specialization shall not transmit any APDU larger than N_{tx} and shall be capable of receiving any APDU up to a size of N_{rx}. For this standard, N_{tx} shall be 7168 octets, and N_{rx} shall be 224 octets.

For an insulin pump agent implementing functions from other device specializations, an upper bound estimation of the APDU sizes brings the following: An agent shall not transmit any APDU larger than the

sum of N_{tx} of all the device specializations implemented and shall be capable of receiving any APDU up to the sum of N_{rx} of all the device specializations implemented. If these numbers are higher than the maximum size determined in ISO/IEEE 11073-20601:2016, the latter shall be applied.

In case the APDU size limit does not allow for the inclusion of a certain amount of multiple pending measurements at the agent, they shall be sent using multiple event reports. See 8.5.3 for the maximum number of measurements allowed for inclusion in a single event report.

8.3 Association procedure

8.3.1 General

Unless otherwise stated, the association procedure for an insulin pump agent and manager according to this standard shall be pursued as specified in ISO/IEEE 11073-20601:2016.

8.3.2 Agent procedure—association request

In the association request sent by the agent to the manager:

- The version of the association procedure used by the agent shall be set to assoc-version1 (i.e., *assoc-version* = 0x80000000).
- The DataProtoList structure element of the data protocol identifier shall be set to data-proto-id-20601 (i.e., *data-proto-id* = 0x5079).
- The *data-proto-info* field shall contain a PhdAssociationInformation structure that shall contain the following parameter values:
  - The agent shall support protocol-version3. Support for any other version may be indicated by setting additional bits. When protocols higher than protocol-version3 are used, the agent shall continue to use only features as specified in this standard. When protocols lower than protocol-version3 are used, the agent shall use only features in that protocol.
  - At least the MDER shall be supported (i.e., *encoding-rules* = 0x8000).
  - The version of the nomenclature used shall be set to nom-version1 (i.e., *nomenclature-version* = 0x80000000).
  - The field *functional-units* may have the test association bits set but shall not have any other bits set.
  - The field *system-type* shall be set to sys-type-agent (i.e., *system-type* = 0x00800000).
  - The *system-id* field shall be set to the value of the System-Id attribute of the MDS object of the agent. The manager may use this field to determine the identity of the insulin pump with which it is associating and, optionally, to implement a simple access restriction policy.
  - The *dev-config-id* field shall be set to the value of the Dev-Configuration-Id attribute of the MDS object of the agent.
  - If the agent supports only the insulin pump specialization, then the field indicating the data request modes (*data-req-mode-capab*) supported by the insulin pump agent shall be set to *data-req-supp-init-agent.*
  - If the agent supports only the insulin pump specialization, then *data-req-init-manager-count* shall be set to zero, and *data-req-init-agent-count* shall be set to 1.

8.3.3 Manager procedure—association response

In the association response message sent by the manager:

- The *result* field shall be set to an appropriate response from those defined in ISO/IEEE 11073-20601:2016. For example, if all other conditions of the association protocol are satisfied, *accepted* is returned when the manager recognizes the *dev-config-id* of the agent and *accepted-unknown-config* otherwise.
- In the DataProtoList structure element, the data protocol identifier shall be set to data-proto-id-20601 (i.e., *data-proto-id* = 0x5079).
- The *data-proto-info* field shall be filled in with a PhdAssociationInformation structure that shall contain the following parameter values:
  - The manager following this specialization shall support protocol-version3. The manager may support additional protocol versions and select them if the agent offers them. When protocols higher than protocol-version3 are used, the manager shall continue to use only features as specified in this standard. When protocols lower than protocol-version3 are used, the manager shall use only features in that protocol.
  - The manager shall respond with a single selected encoding rule that is supported by both agent and manager. The manager shall support at least the MDER.
  - The version of the nomenclature used shall be set to nom-version1 (i.e., *nomenclature-version* = 0x80000000).
  - The field *functional-units* shall have all bits reset except for those relating to a test association.
  - The field *system-type* shall be set to sys-type-manager (i.e., *system-type* = 0x80000000).
  - The *system-id* field shall contain the unique system ID of the manager device, which shall be a valid EUI-64 type identifier.
  - The field *dev-config-id* shall be manager-config-response (0).
  - The field *data-req-mode-capab* shall be 0.
  - If the agent supports only the insulin pump specialization, *data-req-initagent-count* shall be 1 and *data-req-init-manager-count* shall be 0.

8.4 Configuring procedure

8.4.1 General

The agent enters the Configuring state if it receives an association response of accepted-unknown-config. In this case, the configuration procedure as specified in ISO/IEEE 11073-20601:2016 shall be followed. Subclause 8.4.2 specifies the configuration notification and response messages for insulin pump agent with standard configuration ID 1900 (0x076C). Normally, a manager would already know the standard configuration. However, for the purposes of this example, it does not.

8.4.2 Insulin pump—standard configuration

8.4.2.1 Agent procedure

The agent performs the configuration procedure using a “Remote Operation Invoke | Confirmed Event Report” message with an MDC_NOTI_CONFIG event to send its configuration to the manager (see ISO/IEEE 11073-20601:2016). The ConfigReport structure is used for the *event-info* field (see Table 3).

For an insulin pump agent with standard configuration ID 1900 (0x076C), the format and contents of the configuration notification message are as follows:

```
0xE7 0x00                 APDU CHOICE Type (PrstApdu)
0x00 0x70                 CHOICE.length = 112
0x00 0x6e                 OCTET STRING.length = 110
0x00 0x02                 invoke-id = 2 (start of DataApdu. MDER encoded.)
0x01 0x01                 CHOICE(Remote Operation Invoke | Confirmed Event Report)
0x00 0x68                 CHOICE.length = 104
0x00 0x00                 obj-handle = 0 (MDS object)
0x00 0x00 0x00 0x00       event-time = 0
0x0D 0x1C                 event-type = MDC_NOTI_CONFIG
0x00 0x2e                 event-info.length = 46 (start of ConfigReport)
0x07 0x6C                 config-report-id (Dev-Configuration-Id value)
0x00 0x01                 config-obj-list.count = 2 Measurement objects will be "announced"
0x00 0x58                 config-obj-list.length = 88
0x00 0x06                 obj-class = MDC_MOC_VMO_METRIC_NU
0x00 0x01                 obj-handle = 1     (→ 1st object is bolus delivered)
0x00 0x04                 attributes.count = 4
0x00 0x24                 attributes.length = 36
0x09 0x2F                 attribute-id = MDC_ATTR_ID_TYPE
0x00 0x04                 attribute-value.length = 4
0x00 0x80 0x74 0x1C       MDC_PART_PHD_DM | MDC_INS_BOLUS
0x0A 0x46                 attribute-id = MDC_ATTR_METRIC_SPEC_SMALL
0x00 0x02                 attribute-value.length = 2
0x40 0x40                 Metric-Spec-Small (mss-avail-stored-data, mss-acc-agent-initiated)
0x09 0x96                 attribute-id = MDC_ATTR_UNIT_CODE
0x00 0x02                 attribute-value.length = 2
0x15 0x60                 MDC_DIM_INTL_UNIT
0x0A 0x55                 attribute-id = MDC_ATTR_ATTRIBUTE_VAL_MAP
0x00 0x0C                 attribute-value.length = 12
0x00 0x02                 AttrValMap.count = 2
0x00 0x08                 AttrValMap.length = 8
0x0A 0x4C 0x00 0x02       MDC_ATTR_NU_VAL_OBS_BASIC, 2
0x0A 0x82 0x00 0x04       MDC_ATTR_TIME_STAMP_BO, 8
0x00 0x06                 obj-class = MDC_MOC_VMO_METRIC_NU
0x00 0x02                 obj-handle = 2 (→ 2nd measurement is current basal rate setting)
0x00 0x04                 attributes.count = 4
0x00 0x24                 attributes.length = 36
0x09 0x2F                 attribute-id = MDC_ATTR_ID_TYPE
0x00 0x04                 attribute-value.length = 4
0x00 0x80 0x73 0xFC       MDC_PART_PHD_DM | MDC_INS_BASAL_RATE_SET
0x0A 0x46                 attribute-id = MDC_ATTR_METRIC_SPEC_SMALL
0x00 0x02                 attribute-value.length = 2
0x40 0x40                 Metric-Spec-Small (mss-avail-stored-data, mss-acc-agent-initiated)
0x09 0x96                 attribute-id = MDC_ATTR_UNIT_CODE
0x00 0x02                 attribute-value.length = 2
0x16 0x40                 MDC_DIM_INTL_UNIT_PER_HR
0x0A 0x55                 attribute-id = MDC_ATTR_ATTRIBUTE_VAL_MAP
0x00 0x0C                 attribute-value.length = 12
0x00 0x02                 AttrValMap.count = 2
0x00 0x08                 AttrValMap.length = 8
0x0A 0x4C 0x00 0x02       MDC_ATTR_NU_VAL_OBS_BASIC, 2
0x0A 0x82 0x00 0x08       MDC_ATTR_TIME_STAMP_BO, 8
```

8.4.2.2 Manager procedure

The manager shall respond to a configuration notification message using a "Remote Operation Response | Confirmed Event Report" data message with an MDC_NOTI_CONFIG event using the ConfigReportRsp structure for the *event-info* field (see Table 3). As a response to the standard configuration notification message in 8.4.2.1 the format and contents of the manager's configuration notification response message are as follows:

0xE7 0x00	APDU CHOICE Type (PrstApdu)
0x00 0x16	CHOICE.length = 22
0x00 0x14	OCTET STRING.length = 20
0x00 0x02	invoke-id = 0x0002 (mirrored from invocation)
0x02 0x01	CHOICE (Remote Operation Response | Confirmed Event Report)
0x00 0x0E	CHOICE.length = 14
0x00 0x00	obj-handle = 0 (MDS object)
0x00 0x00 0x00 0x00	currentTime = 0
0x0D 0x1C	event-type = MDC_NOTI_CONFIG
0x00 0x04	event-reply-info.length = 4
0x07 0x6C	ConfigReportRsp.config-report-id = 1900
0x00 0x00	ConfigReportRsp.config-result = accepted-config

8.5 Operating procedure

8.5.1 General

Measurement data and status information are communicated from the insulin pump agent during the Operating state. If not stated otherwise, the operating procedure for an insulin pump agent of this standard shall be as specified in ISO/IEEE 11073-20601:2016.

8.5.2 GET insulin pump MDS attributes

See Table 4 for a summary of the GET service.

Refer to the respective standard for the details of getting insulin pump MDS attributes for the specific version of the protocol.

8.5.3 Measurement data transmission

See Table 3 and Table 28 for a summary of the event report services available for measurement data transfer.

To limit the amount of data being transported within an APDU, the insulin pump agent shall not include more than 25 temporarily stored measurements in a single event report. If more than 25 pending measurements are available for transmission, they shall be sent using multiple event reports or using the PM-store mechanism. If multiple insulin pump measurements are available, up to 25 measurements should be transmitted within a single event report. Alternatively, they may be transmitted using a single event report for each insulin pump measurement. However, the former strategy is recommended to reduce overall message size and power consumption.

8.6 Time synchronization

Time synchronization between an insulin pump agent and a manager may be used to coordinate the clocks used when reporting physiological events. Note that the mechanism for synchronizing an agent to a manager is outside the scope of this standard. If time synchronization is used, then this shall be reported in the Mds-Time-Info attribute of the MDS object.

9. Test associations

The test association provides a manufacturer the mechanism to test or demonstrate features of a product in a comprehensive manner. This clause defines the behavior of the standard insulin pump agent during a test association. Support for test association is optional.

9.1 Behavior with standard configuration

An agent or manager entering a test association using the configuration ID for the standard insulin pump device of this standard shall enter the Operating state in test mode. When in test mode, where possible, this should be indicated visually to any user. Normal functionality shall be suspended, and any test data generated shall not be processed by the device as physiological data.

The insulin pump agent shall send a single simulated current basal rate set value of 199.99 IU/h and a single simulated bolus delivery value of 299.99 IU (values never seen in normal usage and outside normal range) within 30 seconds of entering the Operating state. If the measurement-status attribute of the numeric object is implemented, then the test-data bit shall be set.

The test association is terminated in a manner consistent with the agent's normal behavior for terminating an association.

9.2 Behavior with extended configurations

This specification does not define a test association that uses an extended configuration.

10. Conformance

10.1 Applicability

This standard shall be used in conjunction with ISO/IEEE 11073-20601:2016.

An implementation or a system can conform to the following elements of this standard:

— DIM class hierarchy and object definitions (object attributes, notifications, methods, and data type definitions)

— Nomenclature code values

— Protocol and service models

— Communication service model (association and configuration)

10.2 Conformance specification

This standard offers levels of conformance with respect to strict adherence to the standard device and the use of extensions for:

— Information model of a specific device

— Use of attributes, value ranges, and access methods

A vendor shall specify the level of conformance for an implementation based on this standard and provide details of the way in which the definitions of this standard and any extensions are applied.

Specifications shall be provided in the form of a set of implementation conformance statements (ICSs) as detailed in 10.4.

This standard is used in conjunction with ISO/IEEE 11073-20601:2016. It is recommended that the ICS for this standard be created first so that the ICS created for ISO/IEEE 11073-20601:2016 may refer to the ICS for this standard where applicable.

10.3 Levels of conformance

10.3.1 General

This standard defines the following levels of conformance.

10.3.2 Conformance level 1: base conformance

The application uses elements of the information, service, and communication models (object hierarchy, actions, event reports, and data type definitions) and the nomenclature scheme defined in ISO/IEEE 11073-20601:2016 and the ISO/IEEE 11073-104zz documents. All mandatory features defined in the object definition tables and in the ICS tables are implemented. Furthermore, any conditional, recommended, or optional features that are implemented shall follow the requirements in the ISO/IEEE 11073-20601:2016 and ISO/IEEE 11073-104zz documents.

10.3.3 Conformance level 2: extended nomenclature (ASN.1 and/or ISO/IEEE 11073-10101:2004 [B6] and/or ISO/IEEE 11073-10101a:2015 [B7])

Conformance level 2 meets conformance level 1 but also uses or adds extensions in at least one of the information, service, or nomenclature models. Extensions to nomenclature codes shall conform to the ISO/IEEE 11073-10101:2004 [B6] and/or ISO/IEEE 11073-10101a:2015 [B7] framework and lie within the private nomenclature extension range (0xF000 – 0xFFFF).

Extensions to the information or service models shall be fully defined using ASN.1 where appropriate and have their behavior fully described following the framework of the ISO/IEEE 11073-20601:2016 and/or ISO/IEEE 11073-20101:2004 [B9]. All extensions shall be specified and include reference to the definition for the extension, or where no publicly available reference is available, the definition of the extension should be appended to the conformance statement.

10.4 Implementation conformance statements (ICSs)

10.4.1 General format

The ICSs are provided as an overall conformance statement document that comprises a set of tables in the form given by the templates in the following subclauses.

Each ICS table has the following columns:

Index	Feature	Reference	Req/Status	Support	Comment

The table column headings have the following meaning:

— Index: an identifier (e.g., a tag) of a specific feature.

— Feature: briefly describes the characteristic for which a conformance statement is being made.

— Reference: to the clause/paragraph within this document or an external source for the definition of the feature (may be empty).

— Req/Status: specifies the conformance requirement (e.g., mandatory, recommended, etc.)—in some cases, this standard does not specify conformance requirements but requests the status of a particular feature be provided.

— Support: specifies the presence or absence of a feature and any description of the characteristics of the feature in the implementation. This column is to be filled out by the implementer.

— Comment: contains any additional information on the feature. This column is to be filled out by the implementer.

Subclauses 10.4.2 through 10.4.6 specify the format of the specific ICS tables.

10.4.2 General ICS

The general ICS specifies the versions/revisions that are supported by the implementation and high-level system behavior.

Table 37 shows general ICSs.

Table 37—IEEE Std 11073-10419 general ICS table

Index[a]	Feature	Reference	Req./Status	Support	Comment
GEN11073-10419-1	Implementation description	—	Identification of the device/ application. Description of functionality.		
GEN11073-10419-2	Standards followed and their revisions	(standard documents)	(set of existing revisions)	(set of supported revision)	
GEN11073-10419-3	Nomenclature document used and revision	(standard documents)	(set of existing revisions)	(set of supported revisions)	
GEN11073-10419-4	Conformance adherence Level 1	See 10.3.2	Base conformance declaration that device meets the following IEEE Std 11073-10419 conformance requirements: a) All mandatory requirements shall be implemented. b) If implemented, conditional, recommended, and optional requirements shall conform to this standard.	Yes/No (No is not expected as no implies that the implementation is non-conformant)	
GEN11073-10419-5	Conformance adherence Level 2	See 10.3.3	In addition to GEN11073-10419-4, if the device implements extensions and/or additions, they shall conform to nomenclature codes from ASN.1 and/or 10101 framework. These extensions should also be defined in ICS tables pointing toward their reference.	Yes/No	
GEN11073-10419-6	Object containment tree	See 6.3	Provide object containment diagram showing relations between object instances used by the application. A conforming implementation uses only object relations as defined in the DIM.		
GEN11073-10419-7	Nomenclature document used and revision	(standard documents)	(set of existing revisions)	(set of supported revision)	
GEN11073-10419-8	Data structure encoding	—	—	Description of encoding method(s) for ASN.1 data structures	

(Table continues on the next page.)

Table 37—IEEE Std 11073-10419 general ICS table *(continued)*

Index[a]	Feature	Reference	Req./Status	Support	Comment
GEN11073-10419-9	Use of private objects	—	Does the implementation use objects that are not defined in the DIM?	Yes/No (If yes: explain in Table 38)	
GEN11073-10419-10	Use of private nomenclature extensions	—	Does the implementation use private extensions to the nomenclature (i.e., 0xF000–0xFFFF codes from ISO/IEEE 11073-10101:2004 [B6] and/or ISO/IEEE 11073-10101a:2015 [B7])? Private Nomenclature extensions are *only* allowed if the standard nomenclature does not include the specific terms required by the application.	Yes/No (If yes: explain in Table 41)	
GEN11073-10419-11	11073-20601 Conformance		Provide the conformance report required by the ISO/IEEE 11073-20601:2016.		

[a] The prefix GEN11073-10419 is used for the index in the general ICS table.

10.4.3 DIM MOC ICS

The DIM MOC ICS defines which objects are implemented. Information on each object shall be provided as a separate row in the template of Table 38.

Table 38—Template for DIM MOC ICS table

Index	Feature	Reference	Req./Status	Support	Comment
MOC-*n*	Object description	Reference to the clause in the standard or other location where the object is defined.	Implemented	Specify restrictions, e.g., max. number of supported instances	

The *n* in the Index column should be the object handle for implementations that have predefined objects. Otherwise the Index column shall simply be a unique number (1..*m*).

All private objects should be specified and include either a reference to the definition for the object, or where no publicly available reference is available, the definition of the object should be appended to the conformance statement.

The Support column should indicate any restrictions for the object implementation.

An object containment diagram (class instance diagram) should be provided as part of the DIM MOC ICS.

10.4.4 MOC attribute ICS

The MOC attribute ICS defines which attributes, including any inherited attributes, are used/supported in each object of an implementation. Information on each attribute of an object shall be provided as a separate row in the template of Table 39. A separate MOC attribute ICS shall be provided for each object.

Table 39—Template for MOC attribute ICS table

Index	Feature	Reference	Req./Status	Support	Comment
ATTR-*n*-*x*	Attribute name Extended attributes shall include the attribute ID also.	Fill in the reference to the ASN.1 structure if the attribute is not defined in this standard.	M = Mandatory C = Conditional R = Recommended O = Optional (as per definition in attribute definition tables)	Implemented? Yes/No Static/Dynamic Specify restrictions, (e.g., value ranges). Describe how attribute is accessed (e.g., Get, Set, sent in config event report, sent in a data event report). Describe any specific restrictions.	

The Support column shall specify whether the attribute is implemented; for extension attributes, whether the attribute value is static or dynamic; any value ranges; restrictions on attribute access or availability; and any other information.

The *n* in the Index column refers to the ID of the managed object for which the table is supplied (i.e., the index of the managed object as specified in the MOC ICS). There is one separate table for each supported managed object.

The *x* in the Index column is a unique serial number (1..*m*).

10.4.5 MOC notification ICS

The MOC notification ICS specifies all implemented notifications (typically in the form of the event report service) that are emitted by the agent. Table 40 provides a template for use. One table has to be provided for each object that supports special object notifications. One row of the table shall be used for each notification.

Table 40—Template for MOC notification ICS table

Index	Feature	Reference	Req./Status	Support	Comment
NOTI-*n*-*x*	Notification Name and Notification ID	Reference to the clause in the standard or other location where the event is defined.		The Support column shall specify how the notification is sent and any restrictions.	

The *n* in the Index column refers to the ID of the managed object for which the table is supplied (i.e., the index of the managed object as specified in the MOC ICS). There is one separate table for each managed object that supports specific object notifications (i.e., events).

The *x* in the Index column is a unique serial number (1..*m*).

All private notifications should be specified and include reference to the definition for the notification. Where no publicly available reference is available, the definition of the notification should be appended to the conformance statement.

10.4.6 MOC nomenclature ICS

The MOC nomenclature ICS specifies all nonstandard nomenclature codes that are utilized by the agent. Table 41 provides a template for use. One row of the table is to be used for each nomenclature element.

Table 41—Template for MOC nomenclature ICS table

Index	Feature	Reference	Req./Status	Support	Comment
NOME-*n*	Nomenclature Name and Nomenclature value	Reference to the clause in the standard or other location where the nomenclature is defined or used		Describe how the nomenclature is used. Describe any specific restrictions.	

The *n* in the Index column is a unique serial number (1..*m*).

Annex A

(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.

[B1] IEC 60601-1:2005, Ed. 3, Medical electrical equipment—Part 1: General requirements for basic safety and essential performance.[7]

[B2] IEC 60601-2, Medical electrical equipment—Part 2: Particular requirements for the basic safety and essential performance for specific device. [See the entire series of standards, Part 2-1 through Part 2-51.]

[B3] IEC 62304:2006/EN 62304:2006, Medical device software—Software life-cycle processes.[8]

[B4] IEC 80001-1:2010, Application of risk management for IT-networks incorporating medical devices—Part 1: Roles, responsibilities, and activities.

[B5] ISO 14971:2007, Medical devices—Application of risk management to medical devices.[9]

[B6] ISO/IEEE 11073-10101:2004, Health informatics—Point-of-care medical device communication—Part 10101: Nomenclature.[10]

[B7] ISO/IEEE 11073-10101a:2015, Health informatics—Point-of-care medical device communication—Part 10101: Nomenclature Amendment 1: Additional Definitions.

[B8] ISO/IEEE 11073-10201:2004, Health informatics—Point-of-care medical device communication—Part 10201: Domain information model.

[B9] ISO/IEEE 11073-20101:2004, Health informatics—Point-of-care medical device communication—Part 20101: Application profile—Base standard.

[B10] ITU-T Rec. X.680-2002, Information technology—Abstract Syntax Notation One (ASN.1): Specification of basic notation.[11]

[B11] Miles A. A., M. V. Musset, and W. L. M. Perry, “Third international standard for insulin.” *Bulletin of the World Health Organization,* vol. 7, pp. 445–459, 1952. Available: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2554140/pdf/bullwho00630-0071.pdf.

[7] IEC publications are available from the International Electrotechnical Commission (http://www.iec.ch/). IEC publications are also available in the United States from the American National Standards Institute (http://www.ansi.org/).
[8] EN publications are available from the European Committee for Standardization (CEN) (http://www.cen.eu).
[9] ISO publications are available from the ISO Central Secretariat (http://www.iso.ch/). ISO publications are also available in the United States from the American National Standards Institute (http://www.ansi.org/).
[10] ISO/IEEE publications are available from the ISO Central Secretariat (http://www.iso.ch/). ISO/IEEE publications are also available in the United States from The Institute of Electrical and Electronics Engineers (http://standards.ieee.org/).
[11] ITU publications are available from the International Telecommunications Union (http://www.itu.in/).

Annex B

(normative)

Any additional ASN.1 definitions

B.1 Device status and insulin pump status bit mapping

The extension to the enumeration class for operational status requires the following ASN.1 structure definition:

```
OpStat ::= BITS-16 {
        insulin-device-op-undetermined (0),
        insulin-device-op-off (1),
        insulin-device-op-standby (2),
        insulin-device-op-preparing (3),
        insulin-device-op-priming (4),
        insulin-device-op-waiting (5),
        insulin-device-op-ready (6),
        -- reserved for future extension (7),
        -- reserved for future extension (8),
        insulin-device-therapy-stop (9),
        insulin-device-therapy-pause (10),
        insulin-device-therapy-run (11)
        -- reserved for future extension (12),
        -- reserved for future extension (13),
        -- reserved for future extension (14),
        -- reserved for future extension (15),

}
```

The extension to the enumeration class for PHD DM status requires the following ASN.1 structure definition:

```
PHDDMStat ::= BITS-32 {
        device-status-undetermined (0),
        device-status-reset (1),
        -- reserved for future extension (2),
        -- reserved for future extension (3),
        -- reserved for future extension (4),
        device-status-error (5),
        device-status-error-mechanical (6),
        device-status-error-electronic (7),
        device-status-error-software (8),
        device-status-error-battery (9),
        -- reserved for future extension (10),
        -- reserved for future extension (11),
        -- reserved for future extension (12),
        -- reserved for future extension (13),
        -- reserved for future extension (14),
        device-status-service (15),
        device-status-service-time-sync-required (16)
        device-status-service-calibration-required (17),
        device-status-service-replenishment-required (18),
```

```
        -- reserved for future extension (19),
        -- reserved for future extension (20),
        -- reserved for future extension (21),
        -- reserved for future extension (22),
        -- reserved for future extension (23),
        -- reserved for future extension (24),
        device-status-battery-low (25),
        device-status-battery-depleted (26),
        device-status-battery-replaced (27),
        device-status-battery-interrupted (28)
        -- reserved for future extension (29),
        -- reserved for future extension (30),
        -- reserved for future extension (31),
}
```

The extension to the enumeration class for insulin pump status requires the following ASN.1 structure definition.

```
InsPumpStat ::= BITS-32 {
                air-pressure-out-of-range (0)
                bolus-canceled (1)
                delivery-max (2)
                infusion-set-detached (3)
                infusion-set-incomplete (4)
                occlusion-detected (5)
                power-insufficient (6)
                priming-issue (7)
                reservoir-empty (8)
                reservoir-issue (9)
                reservoir-low (10)
                reservoir-attached (11)
                temp-basal-canceled (12)
                temp-basal-expired (13)
                temperature-out-of-range (14)
                }
```

B.2 Capability-mask

-- The capability mask defines if the corresponding bit in the enumeration bit string is supported.

```
CapabMaskSimp ::= BITS-32 {
                bit_0_supported(0),          -- bit is supported: 1 | bit is not supported: 0
                bit_1_supported(1),          -- bit is supported: 1 | bit is not supported: 0
                bit_2_supported(2),          -- bit is supported: 1 | bit is not supported: 0
                bit_3_supported(3),          -- bit is supported: 1 | bit is not supported: 0
                bit_4_supported(4),          -- bit is supported: 1 | bit is not supported: 0
                bit_5_supported(5),          -- bit is supported: 1 | bit is not supported: 0
                bit_6_supported(6),          -- bit is supported: 1 | bit is not supported: 0
                bit_7_supported(7),          -- bit is supported: 1 | bit is not supported: 0
                bit_8_supported(8),          -- bit is supported: 1 | bit is not supported: 0
                bit_9_supported(9),          -- bit is supported: 1 | bit is not supported: 0
                bit_10_supported(10),        -- bit is supported: 1 | bit is not supported: 0
                bit_11_supported(11),        -- bit is supported: 1 | bit is not supported: 0
                bit_12_supported(12),        -- bit is supported: 1 | bit is not supported: 0
                bit_13_supported(13),        -- bit is supported: 1 | bit is not supported: 0
                bit_14_supported(14),        -- bit is supported: 1 | bit is not supported: 0
```

```
                bit_15_supported(15),          -- bit is supported: 1 | bit is not supported: 0
                bit_16_supported(16),          -- bit is supported: 1 | bit is not supported: 0
                bit_17_supported(17),          -- bit is supported: 1 | bit is not supported: 0
                bit_18_supported(18),          -- bit is supported: 1 | bit is not supported: 0
                bit_19_supported(19),          -- bit is supported: 1 | bit is not supported: 0
                bit_20_supported(20),          -- bit is supported: 1 | bit is not supported: 0
                bit_21_supported(21),          -- bit is supported: 1 | bit is not supported: 0
                bit_22_supported(22),          -- bit is supported: 1 | bit is not supported: 0
                bit_23_supported(23),          -- bit is supported: 1 | bit is not supported: 0
                bit_24_supported(24),          -- bit is supported: 1 | bit is not supported: 0
                bit_25_supported(25),          -- bit is supported: 1 | bit is not supported: 0
                bit_26_supported(26),          -- bit is supported: 1 | bit is not supported: 0
                bit_27_supported(27),          -- bit is supported: 1 | bit is not supported: 0
                bit_28_supported(28),          -- bit is supported: 1 | bit is not supported: 0
                bit_29_supported(29),          -- bit is supported: 1 | bit is not supported: 0
                bit_30_supported(30),          -- bit is supported: 1 | bit is not supported: 0
                bit_31_supported(31)           -- bit is supported: 1 | bit is not supported: 0
}

CapabMaskBasic ::=BITS-16 {
                bit_0_supported(0),            -- bit is supported: 1 | bit is not supported: 0
                bit_1_supported(1),            -- bit is supported: 1 | bit is not supported: 0
                bit_2_supported(2),            -- bit is supported: 1 | bit is not supported: 0
                bit_3_supported(3),            -- bit is supported: 1 | bit is not supported: 0
                bit_4_supported(4),            -- bit is supported: 1 | bit is not supported: 0
                bit_5_supported(5),            -- bit is supported: 1 | bit is not supported: 0
                bit_6_supported(6),            -- bit is supported: 1 | bit is not supported: 0
                bit_7_supported(7),            -- bit is supported: 1 | bit is not supported: 0
                bit_8_supported(8),            -- bit is supported: 1 | bit is not supported: 0
                bit_9_supported(9),            -- bit is supported: 1 | bit is not supported: 0
                bit_10_supported(10),          -- bit is supported: 1 | bit is not supported: 0
                bit_11_supported(11),          -- bit is supported: 1 | bit is not supported: 0
                bit_12_supported(12),          -- bit is supported: 1 | bit is not supported: 0
                bit_13_supported(13),          -- bit is supported: 1 | bit is not supported: 0
                bit_14_supported(14),          -- bit is supported: 1 | bit is not supported: 0
                bit_15_supported(15)           -- bit is supported: 1 | bit is not supported: 0
}
```

B.3 State-flag

```
-- The state flag defines if the corresponding bit in the enumeration bit string is a state or an event. For a bit
-- that is a state, then the value is set to 1.

StateFlagSimp ::=BITS-32 {
                bit_0_state(0),     -- bit is state: 1 | bit is event: 0
                bit_1_state(1),     -- bit is state: 1 | bit is event: 0
                bit_2_state(2),     -- bit is state: 1 | bit is event: 0
                bit_3_state(3),     -- bit is state: 1 | bit is event: 0
                bit_4_state(4),     -- bit is state: 1 | bit is event: 0
                bit_5_state(5),     -- bit is state: 1 | bit is event: 0
                bit_6_state(6),     -- bit is state: 1 | bit is event: 0
                bit_7_state(7),     -- bit is state: 1 | bit is event: 0
                bit_8_state(8),     -- bit is state: 1 | bit is event: 0
                bit_9_state(9),     -- bit is state: 1 | bit is event: 0
                bit_10_state(10),   -- bit is state: 1 | bit is event: 0
                bit_11_state(11),   -- bit is state: 1 | bit is event: 0
                bit_12_state(12),   -- bit is state: 1 | bit is event: 0
```

```
                bit_13_state(13),  -- bit is state: 1 | bit is event: 0
                bit_14_state(14),  -- bit is state: 1 | bit is event: 0
                bit_15_state(15),  -- bit is state: 1 | bit is event: 0
                bit_16_state(16),  -- bit is state: 1 | bit is event: 0
                bit_17_state(17),  -- bit is state: 1 | bit is event: 0
                bit_18_state(18),  -- bit is state: 1 | bit is event: 0
                bit_19_state(19),  -- bit is state: 1 | bit is event: 0
                bit_20_state(20),  -- bit is state: 1 | bit is event: 0
                bit_21_state(21),  -- bit is state: 1 | bit is event: 0
                bit_22_state(22),  -- bit is state: 1 | bit is event: 0
                bit_23_state(23),  -- bit is state: 1 | bit is event: 0
                bit_24_state(24),  -- bit is state: 1 | bit is event: 0
                bit_25_state(25),  -- bit is state: 1 | bit is event: 0
                bit_26_state(26),  -- bit is state: 1 | bit is event: 0
                bit_27_state(27),  -- bit is state: 1 | bit is event: 0
                bit_28_state(28),  -- bit is state: 1 | bit is event: 0
                bit_29_state(29),  -- bit is state: 1 | bit is event: 0
                bit_30_state(30),  -- bit is state: 1 | bit is event: 0
                bit_31_state(31)   -- bit is state: 1 | bit is event: 0
}

StateFlagBasic ::=BITS-16 {
                bit_0_state(0),    -- bit is state: 1 | bit is event: 0
                bit_1_state(1),    -- bit is state: 1 | bit is event: 0
                bit_2_state(2),    -- bit is state: 1 | bit is event: 0
                bit_3_state(3),    -- bit is state: 1 | bit is event: 0
                bit_4_state(4),    -- bit is state: 1 | bit is event: 0
                bit_5_state(5),    -- bit is state: 1 | bit is event: 0
                bit_6_state(6),    -- bit is state: 1 | bit is event: 0
                bit_7_state(7),    -- bit is state: 1 | bit is event: 0
                bit_8_state(8),    -- bit is state: 1 | bit is event: 0
                bit_9_state(9),    -- bit is state: 1 | bit is event: 0
                bit_10_state(10), -- bit is state: 1 | bit is event: 0
                bit_11_state(11), -- bit is state: 1 | bit is event: 0
                bit_12_state(12), -- bit is state: 1 | bit is event: 0
                bit_13_state(13), -- bit is state: 1 | bit is event: 0
                bit_14_state(14), -- bit is state: 1 | bit is event: 0
                bit_15_state(15)  -- bit is state: 1 | bit is event: 0
}
```

Annex C

(normative)

Allocation of identifiers

C.1 General

This annex contains the nomenclature codes used in this document and not found in ISO/IEEE 11073-20601:2016. For those not contained in this annex, the normative definition is found in ISO/IEEE 11073-20601:2016.

C.2 Definitions of terms and codes

The format used here follows that of ISO/IEEE 11073-10101:2004 [B6].

```
/*****************************************************************************
* From Communication Infrastructure (MDC_PART_INFRA)
*****************************************************************************/
#define MDC_DEV_SPEC_PROFILE_INSULIN_PUMP 4115  /* Insulin pump              */

/*****************************************************************************
* From Object Infrastructure (MDC_PART_OBJ)
*****************************************************************************/
#define MDC_MOC_VMO_SCHEDSTORE            81    /*                           */
#define MDC_MOC_SCHEDULE_SEG              82    /*                           */
#define MDC_ATTR_SCHED_STORE_HANDLE       2800  /*                           */
#define MDC_ATTR_SCHED_STORE_ACTIVE_INSTNO
                                          2801  /*                           */
#define MDC_ATTR_SCHED_STORE_UPDATED_INSTNO
                                          2802  /*                           */
#define MDC_ATTR_SCHED_STORE_CAPAB        2803  /*                           */
#define MDC_ATTR_SCHED_STORE_CAPAC_CNT    2804  /*                           */
#define MDC_ATTR_SCHED_STORE_USAGE_CNT    2805  /*                           */
#define MDC_ATTR_SCHED_STORE_OP_STAT      2806  /*                           */
#define MDC_ATTR_SCHED_STORE_LABEL_STRING
                                          2807  /*                           */
#define MDC_ATTR_SCHED_SEG_NUM            2816  /*                           */
#define MDC_ATTR_SCHED_SEG_INSTNO         2817  /*                           */
#define MDC_ATTR_SCHED_SEG_MAP            2818  /*                           */
#define MDC_ATTR_SCHED_SEG_PERIOD         2819  /*                           */
#define MDC_ATTR_SCHED_SEG_ENTRY_INTERVAL
                                          2820  /*                           */
#define MDC_ATTR_SCHED_SEG_PERSON_ID      2821  /*                           */
#define MDC_ATTR_SCHED_SEG_ENTRY_CNT      2822  /*                           */
#define MDC_ATTR_SCHED_SEG_LABEL_STRING   2823  /*                           */
#define MDC_ATTR_SCHED_SEG_LAST_UPDATED_ABS_TIME
                                          2828  /*                           */
#define MDC_ATTR_SCHED_SEG_LAST_UPDATED_HIRES_TIME
                                          2829  /*                           */
#define MDC_ATTR_SCHED_SEG_LAST_UPDATED_BO_TIME
                                          2830  /*                           */
#define MDC_ATTR_SCHED_SEG_REF_ABS_TIME   2831  /*                           */
```

```
#define MDC_ATTR_SCHED_SEG_REF_BO_TIME     2832  /*                                */
#define MDC_ATTR_SCHED_SEG_START_ABS_TIME
                                           2833  /*                                */
#define MDC_ATTR_SCHED_SEG_START_BO_TIME
                                           2834  /*                                */
#define MDC_ATTR_SCHED_SEG_END_ABS_TIME    2835  /*                                */
#define MDC_ATTR_SCHED_SEG_END_BO_TIME     2836  /*                                */
#define MDC_ATTR_SCHED_SEG_FIXED_DATA      2840  /*                                */
#define MDC_ATTR_SCHED_SEG_CONFIRM_TIMEOUT
                                           2841  /*                                */
#define MDC_ATTR_SCHED_SEG_TRANSFER_TIMEOUT
                                           2842  /*                                */
#define MDC_ACT_SCHED_SEG_GET_INFO         3108  /*                                */
#define MDC_ACT_SCHED_SEG_GET_ID_LIST      3109  /*                                */
#define MDC_ACT_SCHED_SEG_TRIG_XFER        3110  /*                                */
#define MDC_ACT_SCHED_SEG_DATA             3111  /*                                */
#define MDC_ATTR_ENUM_CAPABILITY_MASK_SIMPLE
                                           2704  /*                                */
#define MDC_ATTR_ENUM_CAPABILITY_MASK_BASIC
                                           2705  /*                                */
#define MDC_ATTR_ENUM_STATE_FLAG_SIMPLE    2706  /*                                */
#define MDC_ATTR_ENUM_STATE_FLAG_BASIC     2707  /*                                */


/************************************************************************************
* From Medical supervisory control and data acquisition (MDC_PART_SCADA)
************************************************************************************/

/************************************************************************************
* From Personal Health Device Disease Management (MDC_PART_PHD_DM)
************************************************************************************/
#define MDC_INS_BASAL                      29680 /* Delivered basal insulin          */
#define MDC_INS_BASAL_RATE_SET             29692 /* Current basal insulin rate setting */
#define MDC_INS_BASAL_PRGM                 29693 /* Programmed basal insulin rate    */
#define MDC_INS_BASAL_TEMP_ABS             29694 /* Temporary basal rate, absolute   */
#define MDC_INS_BASAL_TEMP_REL             29695 /* Temporary basal rate, relative   */
#define MDC_INS_BASAL_UNDETERMINED         29696 /* Undetermined basal               */
#define MDC_INS_BASAL_DEVICE               29697 /* Insulin pump device set basal    */
#define MDC_INS_BASAL_REMOTE               29698 /* Remote control set basal         */
#define MDC_INS_BASAL_AP_CTRL              29699 /* Controller set basal insulin rate */
#define MDC_INS_BASAL_OTHER                29700 /* Rate set by an other source      */

#define MDC_INS_BASAL_RATE_SCHED           29712 /* Basal rate schedule setting      */

#define MDC_INS_BOLUS_SET                  29724 /* Bolus amount set                 */
#define MDC_INS_BOLUS                      29736 /* Delivered bolus insulin          */
#define MDC_INS_BOLUS_FAST                 29737 /* Fast bolus                       */
#define MDC_INS_BOLUS_EXT                  29738 /* Extended bolus                   */
#define MDC_INS_BOLUS_CORR                 29739 /* Correction bolus                 */
#define MDC_INS_BOLUS_MEAL                 29740 /* Meal bolus                       */
#define MDC_INS_BOLUS_UNDETERMINED         29741 /* Undetermined bolus               */

#define MDC_INS_BOLUS_MANUAL               29742 /* Manual, user defined bolus       */
#define MDC_INS_BOLUS_RECOMMENDED          29743 /* Recommended bolus                */
#define MDC_INS_BOLUS_MANUAL_CHANGE        29744 /* Recommended bolus changed by
                                                    user                            */
#define MDC_INS_BOLUS_COMMANDED            29745 /* Commanded bolus                  */
```

```
#define MDC_INS_BOLUS_OTHER                 29746 /* Other bolus                    */
#define MDC_INS_BOLUS_PENDING_DELAY         29747 /* Bolus pending delay            */

#define MDC_INS_I2CHO_SCHED                 29756 /* I:CHO Schedule Setting         */
#define MDC_INS_ISF_SCHED                   29768 /* ISF Schedule Setting           */
#define MDC_INS_RESERVOIR                   29780 /* Insulin reservoir remaining    */
#define MDC_INS_CONC                        29792 /* Insulin concentration          */

#define MDC_INS_PUMP_OP_STAT                29804 /* Operational status             */
#define MDC_PHD_DM_DEV_STAT                 20000 /* PHD DM Device status           */
#define MDC_INS_PUMP_DEV_STAT               29836 /* Insulin Pump Device status     */

/********************************************************************************
* From Dimensions (MDC_PART_DIM)
********************************************************************************/
#define MDC_DIM_DIMLESS                     512   /* Dimensionless                  */
#define MDC_DIM_G                           1728  /* Gram                           */
#define MDC_DIM_INTL_UNIT                   5472  /* International units            */
#define MDC_DIM_INTL_UNIT_PER_HR            5696  /* International units per hour   */
#define MDC_DIM_INTL_UNIT_PER_L             5568  /* International units per liter  */
#define MDC_DIM_INTL_UNIT_PER_ML            5600  /* International units per milliter */
#define MDC_DIM_INTL_UNIT_PER_M_CUBE        5536  /* International units
                                                     per cubic meter                */
#define MDC_DIM_INTL_UNIT_PER_CM_CUBE       5504  /* International units
                                                     per cubic centimeter           */
#define MDC_DIM_MILLI_G_PER_DL              2130  /* Milligram per deciliter        */
#define MDC_DIM_MILLI_MOLE_PER_L            4722  /* Millimole per liter            */
```

C.3 Systematic derivations of terms and codes

Systematic derivations of terms and codes are outlined in Table C.1.

Table C.1—Systematic derivations of terms and codes

Systematic name	Common term	Acronym	Description/definition	Reference ID	Code
Disease Management | Device Status | Personal Health Device	PHD DM Status		Object containing the general device status for PHD disease management.	MDC_PHD_DM_DEV_STAT	20000
Insulin | Basal	Insulin Basal		Object containing the basal delivered amount.	MDC_INS_BASAL	29680
Insulin | Basal | Rate Setting	Insulin Basal Rate Setting		Object containing the current basal rate setting.	MDC_INS_BASAL_RATE_SET	29692
Insulin | Basal | Programmed	Insulin Basal Rate Programmed		Basal rate set by the active basal profile schedule setting.	MDC_INS_BASAL_PRGM	29693
Insulin | Basal | Temporary | Absolute	Absolute Temporary Insulin Basal Rate		Basal rate set by an absolute temporary change.	MDC_INS_BASAL_TEMP_ABS	29694
Insulin | Basal | Temporary | Relative	Relative Temporary Insulin Basal Rate		Basal rate set by an relative temporary change.	MDC_INS_BASAL_TEMP_REL	29695
Insulin | Basal | Undetermined	Undetermined Insulin Basal Rate Context		Undetermined basal rate context.	MDC_INS_BASAL_UNDETERMINED	29696
Insulin | Basal | Device	Insulin Pump Device Set Basal Rate		Basal rate set by the insulin pump device.	MDC_INS_BASAL_DEVICE	29697
Insulin | Basal | Remote	Insulin Pump Remote Control Set Basal Rate		Basal rate set by a remote control.	MDC_INS_BASAL_REMOTE	29698
Insulin | Basal | Artificial Pancreas	Artificial Pancreas Controller Set Basal Rate		Basal rate set by an artificial pancreas controller.	MDC_INS_BASAL_AP_CTRL	29699
Insulin | Basal | Other	Other Source Set Basal Rate		Basal rate set by another source. This option is used when the basal rate source does not match an available option.	MDC_INS_BASAL_OTHER	29670
Insulin | Basal | Rate Schedule	Insulin Basal Rate Segment		Object containing the basal rate segment for a basal rate schedule.	MDC_INS_BASAL_RATE_SCHED	29712
Insulin | Bolus | Setting	Insulin Bolus Setting		Object containing the current bolus setting.	MDC_INS_BOLUS_SET	29724
Insulin | Bolus	Insulin Bolus		Object containing the bolus delivered amount.	MDC_INS_BOLUS	29736

(Table continues on the next page.)

Table C.1—Systematic derivations of terms and codes *(continued)*

Systematic name	Common term	Acronym	Description/definition	Reference ID	Code
Insulin | Bolus | Fast	Insulin Fast Bolus		Insulin delivered as a fast bolus.	MDC_INS_BOLUS_FAST	29737
Insulin | Bolus | Extended	Insulin Extended Bolus		Insulin delivered as an extended bolus.	MDC_INS_BOLUS_EXT	29738
Insulin | Bolus | Correction	Insulin Correction Bolus		Insulin delivered as a bolus for a correction.	MDC_INS_BOLUS_CORR	29739
Insulin | Bolus | Meal	Insulin Meal Bolus		Insulin delivered as a bolus for a meal.	MDC_INS_BOLUS_MEAL	29740
Insulin | Bolus | Undetermined	Undetermined Insulin Bolus Context		Undetermined bolus context.	MDC_INS_BOLUS_UNDETERMINED	29741
Insulin | Bolus | Manual	Manual Bolus		Bolus defined and set manually by a user.	MDC_INS_BOLUS_MANUAL	29742
Insulin | Bolus | Recommended	Recommended Bolus		Bolus recommended to the user such a bolus calculator.	MDC_INS_BOLUS_RECOMMENDED	29743
Insulin | Bolus | Manual Change	Manually Changed Bolus		Bolus recommended to the user such a bolus calculator but bolus amount changed by the user.	MDC_INS_BOLUS_MANUAL_CHANGE	29744
Insulin | Bolus | Commanded	Commanded Bolus		Bolus commanded by another device such as an artificial pancreas controller.	MDC_INS_BOLUS_COMMANDED	29745
Insulin | Bolus | Other	Other Source Set Bolus		Bolus set by another source. This option is used when the bolus source does not match an available option.	MDC_INS_BOLUS_OTHER	29746
Insulin | Bolus | Pending Delay	Bolus Pending Delay		Object containing the bolus pending delay.	MDC_INS_BOLUS_PENDING_DELAY	29747
Insulin | Carbohydrate | Schedule	I:CHO		Object containing the I:CHO for an I:CHO schedule.	MDC_INS_I2CHO_SCHED	29756
Insulin | Sensitivity Factor | Schedule	ISF		Object containing the ISF for an ISF schedule.	MDC_INS_ISF_SCHED	29768
Insulin | Operational Status	Insulin Device Status		Object containing the insulin pump operational status.	MDC_INS_PUMP_OP_STAT	29804
Insulin | Device Status	Insulin Device Status		Object containing the insulin pump device status.	MDC_INS_PUMP_DEV_STAT	29836

(Table continues on the next page.)

Table C.1—Systematic derivations of terms and codes *(continued)*

Systematic name	Common term	Acronym	Description/definition	Reference ID	Code
Attribute | Schedule | Store | Handle	Reference Ideintifier		Schedule-store attribute representing a reference ID for this object. Each object shall have a unique ID assigned by the agent. The handle identifies the object in event reports sent to the manager and to address the object instance in messages invoking object methods.	MDC_ATTR_SCHED_STORE_HANDLE	2800
Attribute | Schedule | Store | Instance Number | Active Segment	Active Schedule Identifier		Schedule-store attribute representing the unique ID of the specific schedule-segment object that is active. If no schedule-segment is currently active or there are currently no schedule-segments, the value shall be 0.	MDC_ATTR_SCHED_STORE_ACTIVE_INSTNO	2801
Attribute | Schedule | Store | Instance Number | Updated List	Updated Schedule Identifier(s)		Schedule-store attribute representing the list of schedule-segment object IDs reporting all updated schedule-segments (i.e., created, modified, or deleted).	MDC_ATTR_SCHED_STORE_UPDATED_INSTNO	2802
Attribute | Schedule | Store | Capabilities	Schedule-Store Capabilities		Schedule-store attribute representing basic capabilities of the schedule-store object instance.	MDC_ATTR_SCHED_STORE_CAPAB	2803
Attribute | Schedule | Store | Count | Capacity	Maximum Capacity Count		Schedule-store attribute representing the maximum number of stored schedule-segment entries (entries in all contained schedule-segments).	MDC_ATTR_SCHED_STORE_CAPAC_CNT	2804
Attribute | Schedule | Store | Count | Usage	Number of Schedule-Segments		Schedule-store attribute representing the actual number of currently stored schedule-segment entries (entries in all contained schedule-segments).	MDC_ATTR_SCHED_STORE_USAGE_CNT	2805
Attribute | Schedule | Store | Operational Status	Schedule-Store Operational Status		Schedule-store attribute indicating if new entries are currently being inserted in any of the contained schedule-segments. If any schedule-segment contained by this schedule-store is being modified, this attribute shall be set to enabled. Otherwise, it shall be set to disabled.	MDC_ATTR_SCHED_STORE_OP_STAT	2806
Attribute | Schedule | Store | Label	Schedule-Store Label		Schedule-store attribute representing an application-dependent label for the schedule-store in printable ASCII to indicate its intended use and may be used for display purposes.	MDC_ATTR_SCHED_STORE_LABEL_STRING	2807

(Table continues on the next page.)

Table C.1—Systematic derivations of terms and codes *(continued)*

Systematic name	Common term	Acronym	Description/definition	Reference ID	Code
Attribute | Schedule | Store | Number of Schedule Segments	Number of Schedule-Segments		Schedule-store representing the number of currently instantiated schedule-segments contained in the schedule-store. Note that the schedule-segment attribute Instance-Number is NOT related to this number (i.e., Instance-Number does not need to be in the range from 0 to number-of-schedule-segments).	MDC_ATTR_SCHED_SEG _NUM	2816
Method | Schedule | Store | Get | Schedule Segment Info	Get Schedule-Segment Info		Schedule-store object method that allows the manager to retrieve Schedule-segment attributes of one or more Schedule-segments, with the exception of the Fixed-Schedule-Segment-Data attribute which contains the actual schedule data and is retrieved by using the Trig-Schedule-Segment-Data-Xfer method.	MDC_ACT_SCHED_SEG_GET_INFO	3108
Method | Schedule | Store | Get | Schedule Segment Data Transfer	Get Schedule-Segment Data Transfer		Schedule-store object method that allows the manager to retrieve a list of the instance numbers of all the schedule-segments of a schedule-store.	MDC_ACT_SCHED_SEG_GET_ID_LIST	3109
Method | Schedule | Store | Trigger | Schedule Segment Info	Trigger Schedule-Segment Info		Schedule-store object method that allows the manager to start the transfer of the Fixed-Schedule-Segment-Data attribute of a specified schedule-segment.	MDC_ACT_SCHED_SEG_TRIG_XFER	3110
Event | Schedule | Store | Schedule Segment Data Event	Schedule-Segment Data Event		Schedule-store object event that sends data stored in the Fixed-Schedule-Segment-Data of a schedule-segment.	MDC_NOTI_SCHED_SEG_DATA	3111
Attribute | Schedule | Segment | Identifier	Schedule-Segment Identifier		Schedule-segment attribute representing the ID of a specific schedule-segment object instance. Each instance shall have a unique number assigned by the agent starting at 1. It is used by the manager to address a schedule-segment. Note, the instance-number value of 0 shall be reserved for reporting no active schedule-segment in the schedule-store.	MDC_ATTR_SCHED_SEG_INSTNO	2817

(Table continues on the next page.)

Table C.1—Systematic derivations of terms and codes *(continued)*

Systematic name	Common term	Acronym	Description/definition	Reference ID	Code
Attribute | Schedule | Segment | Map	Schedule-Segment Map		Schedule-segment attribute representing the format and contents of one schedule entry. An entry has a conditional header containing information applicable to all elements in the entry. If the schedule-segment period, start time and entry interval are used, then the header information shall not be used. The entry then contains one or more elements, defined by the handle and an attribute value map defining the object attributes for each element in the schedule-segment.	MDC_ATTR_SCHED_SEG_MAP	2818
Attribute | Schedule | Segment | Period	Schedule-Segment Period		Schedule-segment attribute representing the period of the schedule-segments. If the schedule-segment occurs only once, then the value shall be 0.	MDC_ATTR_SCHED_SEG_PERIOD	2819
Attribute | Schedule | Segment | Entry Interval	Schedule-Segment Entry Interval		Schedule-segment attribute representing the interval between each entry in schedule-segments.	MDC_ATTR_SCHED_SEG_ENTRY_INTERVAL	2820
Attribute | Schedule | Segment | Person ID	Person Identification		Schedule-segment attribute representing the person ID. This standard supports devices that have simple support for data from multiple persons. A person ID is used to differentiate different persons. If the schedule-segment is able to have schedule data for multiple persons, it shall set the schedsc-multi-person bit in the Sched-Store-Capab attribute. If this bit is set, all schedule-segment instances shall support the Schedule-Segment-Person-Id attribute. Otherwise, this attribute is not defined.	MDC_ATTR_SCHED_SEG_PERSON_ID	2821
Attribute | Schedule | Segment | Count | Entry	Schedule-Segment Entry Count		Schedule-segment attribute representing the actual number of schedule-segment entries.	MDC_ATTR_SCHED_SEG_ENTRY_CNT	2822
Attribute | Schedule | Segment | Label	Schedule-Segment Label		Schedule-segment attribute representing an application-dependent label in printable ASCII for the schedule-segment to indicate its intended use and may be used for display purposes.	MDC_ATTR_SCHED_SEG_LABEL_STRING	2823

(Table continues on the next page.)

Table C.1—Systematic derivations of terms and codes *(continued)*

Systematic name	Common term	Acronym	Description/definition	Reference ID	Code
Attribute | Schedule | Segment | Time | Last Update | Absolute	Last Updated Time (Absolute)		Schedule-segment attribute representing the time the schedule-segment was last updated. If this attribute is used, neither the Schedule-Segment-LastUpdated-HiRes-Time nor Schedule-Segment-LastUpdated-BO-Time shall be used.	MDC_ATTR_SCHED_SEG_LAST_UPDATED_ABS_TIME	2828
Attribute | Schedule | Segment | Time | Last Update | High Resolution	Last Updated Time (High Resolution)		Schedule-segment attribute representing the time the schedule-segment was last updated. If this attribute is used, neither the Schedule-Segment-LastUpdated-Abs-Time nor Schedule-Segment-LastUpdated-BO-Time shall be used.	MDC_ATTR_SCHED_SEG_LAST_UPDATED_HIRES_TIME	2829
Attribute | Schedule | Segment | Time | Last Update | Base Offset	Last Updated Time (Base Offset)		Schedule-segment attribute representing the time the schedule-segment was last updated. If this attribute is used, neither the Schedule-Segment-LastUpdated-Abs-Time nor Schedule-Segment-LastUpdated-HiRes-Time shall be used.	MDC_ATTR_SCHED_SEG_LAST_UPDATED_BO_TIME	2830
Attribute | Schedule | Segment | Time | Reference | Absolute	Reference Time (Absolute)		Schedule-segment attribute representing the time from which all entries within the schedule-segment are referenced. This attribute shall be used, if the Schedule-Segment-Reference-BO-Time is not used.	MDC_ATTR_SCHED_SEG_REF_ABS_TIME	2831
Attribute | Schedule | Segment | Time | Reference | Base Offset	Reference Time (Base Offset)		Schedule-segment attribute representing the time from which all entries within the schedule-segment are referenced. This attribute shall be used, if the Schedule-Segment-Reference-Abs-Time is not used.	MDC_ATTR_SCHED_SEG_REF_BO_TIME	2832
Attribute | Schedule | Segment | Time | Start | Absolute	Start Time (Absolute)		Schedule-segment attribute representing the time of the first execution of the schedule. If this attribute is used, the Schedule-Segment-Start-BO-Time shall not be used. If this attribute is used, the schedule-segment becomes the active-schedule-segment in the schedule-store at the start time. A schedule-segment shall not be active before the start time.	MDC_ATTR_SCHED_SEG_START_ABS_TIME	2833

(Table continues on the next page.)

Table C.1—Systematic derivations of terms and codes *(continued)*

Systematic name	Common term	Acronym	Description/definition	Reference ID	Code
Attribute | Schedule | Segment | Time | End | Absolute	End Time (Absolute)		Schedule-segment attribute representing the expiration of the schedule. If this attribute is used, the Schedule-Segment-End-BO-Time shall not be used. If this attribute is used and the end time is reached, the schedule-segment shall no longer be the active-schedule-segment in the schedule-store.	MDC_ATTR_SCHED_SEG_END_ABS_TIME	2835
Attribute | Schedule | Segment | Time | Start | Base Offset	Start Time (Base Offset)		Schedule-segment attribute representing the time of the first execution of the schedule. If this attribute is used, the Schedule-Segment-Start-Abs-Time shall not be used. If this attribute is used, the schedule-segment becomes the active-schedule-segment in the schedule-store at the start time. A schedule-segment shall not be active before the start time.	MDC_ATTR_SCHED_SEG_START_BO_TIME	2834
Attribute | Schedule | Segment | Time | End | Base Offset	End Time (Base Offset)		Schedule-segment attribute representing the expiration of the schedule. If this attribute is used, the Schedule-Segment-End-Abs-Time shall not be used. If this attribute is used and the end time is reached, the schedule-segment shall no longer be the active-schedule-segment in the schedule-store.	MDC_ATTR_SCHED_SEG_END_BO_TIME	2836
Attribute | Schedule | Segment | Fixed Data	Schedule-Segment Fixed Data		Schedule-segment attribute representing the schedule data transferred as an array of entries in a format as specified in the Schedule-Segment-Entry-Map attribute. This is defined here as an opaque data structure without a defined data type. Note that this attribute is not directly accessible; it is only retrievable by the manager using the Trig-Schedule-Segment-Data-Xfer method.	MDC_ATTR_SCHED_SEG_FIXED_DATA	2840

(Table continues on the next page.)

Table C.1—Systematic derivations of terms and codes *(continued)*

Systematic name	Common term	Acronym	Description/definition	Reference ID	Code
Attribute | Schedule | Segment | Timeout | Confirm	Confirm Timeout		Schedule-segment attribute representing the minimum time that the agent shall wait for a response message from the manager after issuing a Confirmed Event Report invoke message before timing out and transitioning to the Unassociated state. This is an informational attribute for the benefit of the manager. If this attribute is supplied, it shall match the actual timeout value that the agent uses for the Confirmed Event Report generated from the Schedule-store object. This attribute is informational for the manager in the sense that the manager does not use this attribute in an actual implementation of the protocol (i.e., the manager does not timeout on an agent-generated Confirmed Event Report). However, the manager might wish to use this information to prioritize its handling of a "short" timeout agent over that of a "long" timeout agent.	MDC_ATTR_SCHED_SEG_CONFIRM_TIMEOUT	2841
Attribute | Schedule | Segment | Timeout | Transfer	Transfer Timeout		Schedule-segment attribute representing the minimum time that the manager shall wait for the complete transfer of schedule-segment information. If the timeout expires prior to the reception of the complete schedule-segment, the manager shall transition to the Unassociated state as described in 8.9.5.6 of ISO/IEEE 11073-20601:2016.	MDC_ATTR_SCHED_SEG_TRANSFER_TIMEOUT	2842

Annex D

(informative)

Message sequence examples

Figure D.1 shows a sequence diagram of the messaging procedure corresponding to the following use case. The user of an insulin pump agent device intends to connect it to a manager device for the first time. The insulin pump is capable of performing basal rate set and bolus measurements.

a) When the user connects the insulin pump, the manager does not recognize the agent's configuration and sends a response to the agent's association request with the result *accepted-unknown-config*. See I.2.2.2 and I.2.2.3 for the corresponding PDU examples.

b) As a consequence of this, the agent negotiates its configuration information to the manager. After getting confirmation from the manager accepting the agent's configuration, the agent device is ready to send measurements. Both devices enter the Operating state. See I.3.2.2 and I.3.2.3 for the corresponding PDU examples.

c) Subsequently, the manager may request the MDS object attributes of the agent by sending a data message with the "Remote Operation Invoke | Get" command. Note that the manager may request the MDS object attributes as soon as the agent enters the Associated state, including the Configuring and Operating substates. As a response, the agent reports its MDS object attributes to the manager using a Data message with the "Remote Operation Response | Get" command. See I.4.1.2 and I.4.1.3 for the corresponding PDU examples.

d) As a next step, the user of the agent device takes several measurements over some period of time. The measurement data are transmitted to the manager using an unconfirmed event reports. See I.5.1 for the corresponding PDU example.

e) The user ends the measurement session (e.g., by pushing a proper button on the device, or just by not using the device for a duration longer than a certain time period). As a consequence, the agent disassociates from the manager by sending an association release request. The manager responds with an association release response. See I.6.1 and I.6.2 for the corresponding PDU examples.

f) When the agent requests to associate to the manager for the next measurement session (e.g., the next day), the result in the manager's response is *accepted*, as it already knows the agent's configuration from the previous measurement session. Both devices transition directly to the Operating state.

g) Finally, the last two steps shown are similar as in item d) and item e). The user takes several unconfirmed measurements followed by releasing the association.

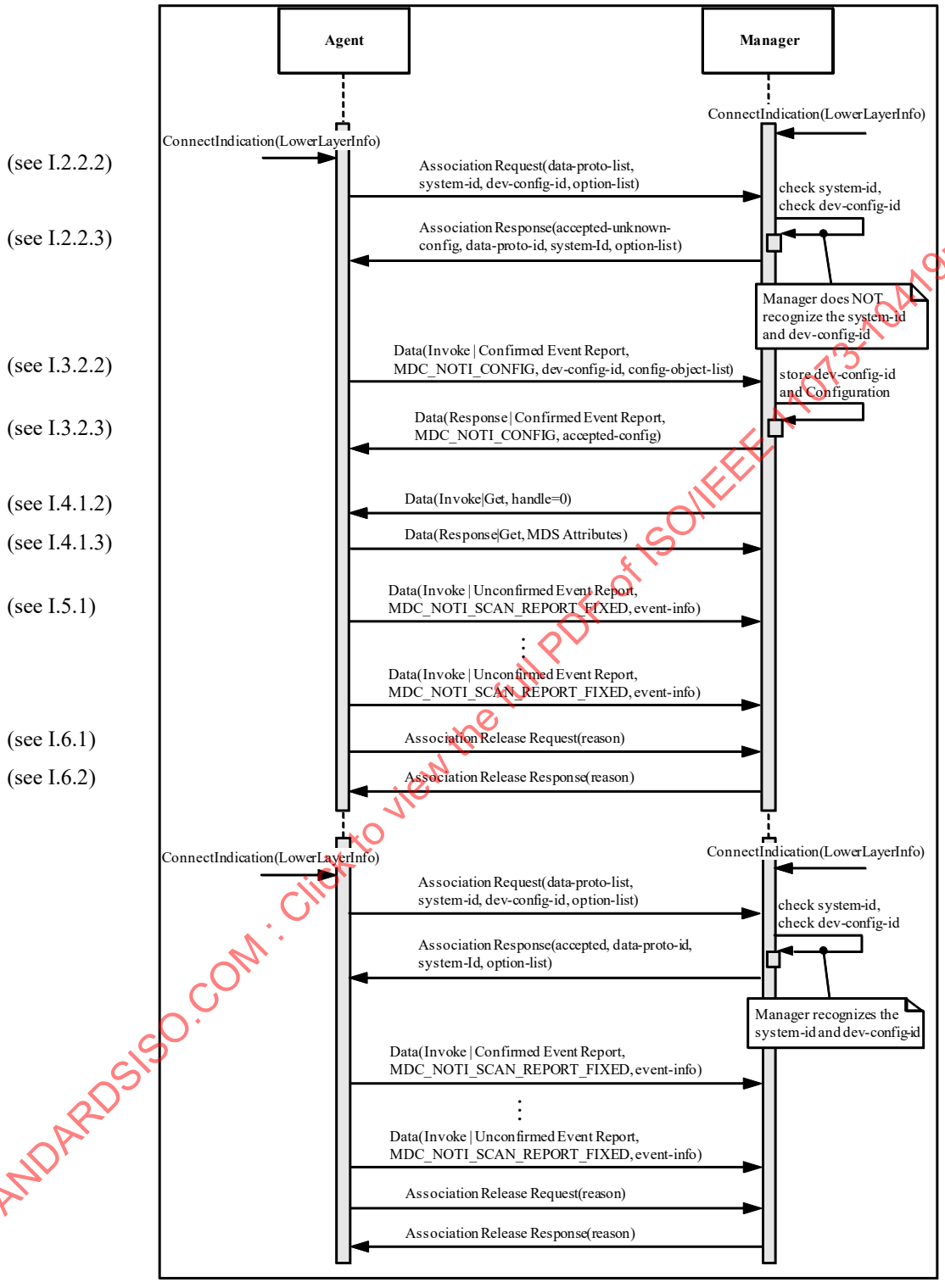

Figure D.1—Sequence diagram for insulin pump example use case

Annex E

(normative)

Schedule-store class

E.1 Schedule-store class

E.1.1 General

An instance of the schedule-store class provides storage capabilities for a series of schedules. Data related to these schedules are stored in a variable number of schedule-segment objects (see E.2). The stored schedule data of the schedule-store object are requested from the agent by the manager using object access services (see 7.2). Anybody not familiar with the schedule-store concept may wish to read Annex G for a conceptual overview prior to reading the following subclauses.

Attribute values stored in a schedule-segment may require the use of additional attributes from that object to describe the schedule; a common example being the duration. If an attribute value in a schedule-segment depends on an attribute value not stored in the schedule-segment, then that dependent attribute shall not change value during the lifetime of the schedule-segment. Otherwise, the agent shall store the dependent attribute value in the schedule-segment.

E.1.2 Schedule-store class identification

The nomenclature code to identify the schedule-store class is MDC_MOC_VMO_SCHEDSTORE.

E.1.3 Schedule-store class attributes

Table E.1 defines the set of schedule-store attributes that are supported for PHD communication.

Table E.1—Schedule-store attributes

Attribute name	Attribute ID	Attribute type	Remark	Qualifiers
Schedule-Handle	MDC_ATTR_SCHED_STORE_HANDLE	HANDLE	The Schedule Handle attribute represents a reference ID for this object. Each object shall have a unique ID assigned by the agent. The handle identifies the object in event reports sent to the manager and to address the object instance in messages invoking object methods.	Mandatory Static
Active-Schedule-Segment-Instance-Number	MDC_ATTR_SCHED_STORE_ACTIVE_INSTNO	InstNumber	The Active-Schedule-Instance-Number is the unique ID of the specific schedule-segment object that is active. If no schedule-segment is currently active or there are currently no schedule-segments, the value shall be 0.	Mandatory Dynamic

Table E.1—Schedule-store attributes *(continued)*

Attribute name	Attribute ID	Attribute type	Remark	Qualifiers
Updated-Schedule-Segment-Instance-Number-List	MDC_ATTR_SCHED_STORE_UPDATED_INSTNO	InstNumberList	The Updated-Schedule-Instance-Number-list is the list of schedule-segment object IDs reporting all updated schedule-segments (i.e., created, modified, or deleted).	Mandatory Observational
Schedule-Store-Capab	MDC_ATTR_SCHED_STORE_CAPAB	SchedStoreCapab	This attribute defines basic capabilities of the schedule-store object instance.	Mandatory Static
Schedule-Store-Capacity-Count	MDC_ATTR_SCHED_STORE_CAPAC_CNT	INT-U32	This attribute is the maximum number of stored schedule-segment entries (entries in all contained schedule-segments).	Optional Static
Schedule-Store-Usage-Count	MDC_ATTR_SCHED_STORE_USAGE_CNT	INT-U32	This attribute is the actual number of currently stored schedule-segment entries (entries in all contained schedule-segments).	Optional Dynamic
Schedule-Store-Operational-Status	MDC_ATTR_SCHED_STORE_OP_STAT	OperationalState	The attribute indicates if new entries are currently being inserted in any of the contained schedule-segments. If any schedule-segment contained by this schedule-store is being modified, this attribute shall be set to enabled. Otherwise, it shall be set to disabled.	Mandatory Dynamic
Schedule-Store-Label	MDC_ATTR_SCHED_STORE_LABEL_STRING	OCTET STRING	This attribute is an application-dependent label for the schedule-store in printable ASCII to indicate its intended use and may be used for display purposes.	Optional Static
Number-Of-Schedule-Segments	MDC_ATTR_SCHED_SEG _NUM	INT-U16	This attribute is the number of currently instantiated schedule-segments contained in the schedule-store. Note that the schedule-segment attribute Instance-Number is NOT related to this number (i.e., Instance-Number does not need to be in the range from 0 to Number-Of-Schedule-Segments).	Mandatory Dynamic

E.1.4 Schedule-store object methods

Table E.2 defines the methods (actions) of a schedule-store object. These methods can be invoked using the ACTION service.

Table E.2—Schedule-store object methods

Method/Action	Mode	Action-type	action-info-args	Resulting action-info-args
Get-Schedule-Segment-Info	Confirmed	MDC_ACT_SCHED_SEG_GET_INFO	SchedSegmSelection	SchedSegmentInfoList
Get-Schedule-Segment-Id-List	Confirmed	MDC_ACT_SCHED_SEG_GET_ID_LIST	(empty)	SchedSegmIdList
Trig-Schedule-Segment-Data-Xfer	Confirmed	MDC_ACT_SCHED_SEG_TRIG_XFER	TrigSchedSegmDataXferReq	TrigSchedSegmDataXferRsp

If an agent supports the schedule-store class, the support of the Get-Schedule-Segment-Info or Get-Schedule-Segment-Id-List methods is mandatory, and support of the Trig-Segment-Data-Xfer method is mandatory.

If a manager supports the schedule-store class, the support of sending the Get-Schedule-Segment-Info, Get-Schedule-Segment-Id-List and Trig-Segment-Data-Xfer methods is mandatory.

— **Get-Schedule-Segment-Info:**

This method allows the manager to retrieve Schedule-segment attributes of one or more schedule-segments, with the exception of the Fixed-Schedule-Segment-Data attribute which contains the actual schedule data and is retrieved by using the Trig-Schedule-Segment-Data-Xfer method. In particular, the Get-Schedule-Segment-Info method allows the manager to retrieve the attributes and their data contents from the Schedule-segment object instances identified by the SchedSegmSelection parameter.

The agent shall support the all-sched-segments choice in the SchedSegmSelection action-info-args of the Get-Schedule-Segment-Info method. The agent may support the sched-segm-id-list choice in the SchedSegmSelection action-info-args of the Get-Schedule-Segment-Info method. In this case the agent shall set the schedsc-segm-id-list-select flag in the Schedule-Store-Capab attribute. If the manager sends the Get-Schedule-Segment-Info method with the choice that the agent does not support, the agent shall reply with an unsupported-choice error (roer).

If the manager supports sending the Get-Schedule-Segment-Info method, the manager shall support at least the choice all-segments in the SchedSegmSelection action-info-args of the Get-Schedule-Segment-Info method. The manager may support additional choices.

If there is no schedule-segment that matches the selection criteria in the SchedSegmSelection action-info-args such that no schedule-segments are found by the action, then this is not an error and a normal response is sent, and the segment info list will just be empty.

If the choice of SchedSegmSelection in the Get-Schedule-Segment-Info method is sched-segm-id-list and the sched-segm-id-list is empty, then the response shall be a sched-segment-info-list that is empty.

If the agent supports the Get-Schedule-Segment-Info method, the agent shall set the schedsc-get-segm-info-sup flag in the Schedule-Store-Capab attribute.

— **Get-Schedule-Segment-Id-List:**

This method allows the manager to retrieve a list of the instance numbers of all the schedule-segments of a schedule-store. In particular, the Get-Schedule-Segm-Id-List method allows the manager to then retrieve the attributes of selected schedule-segment object instances and their data contents without need to retrieve information of all schedule-segments. This also allows the manager to retrieve multiple schedule-segments as a series of requests.

If the agent supports the Get-Schedule-Segment-Id-List method, the agent shall set the schedsc-get-segm-id-list-sup flag in the Schedule-Store-Capab attribute.

If the agent supports the Get-Schedule-Segment-Id-List method, the agent shall support the action of Get-Schedule-Segment-Info by ID list.

— **Trig-Schedule-Segment-Data-Xfer:**

This method allows the manager to start the transfer of the Fixed-Schedule-Segment-Data attribute of a specified schedule-segment. The agent indicates in the response if it accepts or denies this request. If the agent accepts the request, the agent sends Schedule-Segment-Data-Event messages as described in E.1.5. If this method is invoked on a schedule-segment that has the Operational-State attribute set to enabled, the agent shall reply with a not-allowed-by-object error (roer) with a return code of MDC_RET_CODE_OBJ_BUSY.

E.1.5 Schedule-store object events

Table E.3 defines the potential events sent by a schedule-store object.

Table E.3—Schedule-store object events

Event	Mode	Event-type	Event-info parameter	Event-reply-info
Schedule-Segment-Data-Event	Confirmed	MDC_NOTI_SCHED_SEG_DATA	ScheduleSegmentDataEvent	ScheduleSegmentDataResult

— **Schedule-Segment-Data-Event:**

This event sends data stored in the Fixed-Schedule-Segment-Data of a schedule-segment from the agent to the manager. The event is triggered by the manager by the Trig-Schedule-Segment-Data-Xfer method. Once the data transfer is triggered, the agent sends Schedule-Segment-Data-Event messages until the complete Fixed-Schedule-Segment-Data is transferred or the transfer is aborted by the manager or agent. See item c) in H.1.1.2.2.1.1 for a full description of transfer schedule-segment content.

It is encouraged to place as many segment entries contained in a Schedule-Segment-Data-Event as possible to reduce the number of messages required for the transfer of the segment. The agent shall transfer all segment entries in order, first entry first (first in, first out).

Support for the event by the agent is mandatory if the agent supports schedule-store objects.

If confirmed event report is used by a schedule-store object, there shall be at most one unacknowledged confirmed event report outstanding from this object at any point in time.

E.1.6 Other schedule-store services

E.1.6.1 GET service

Support for the GET service shall be provided by any agent that supports one or more schedule-store objects only while in the Operating state. The manager uses the GET service to retrieve the values of all schedule-store object attributes. If a manager does not have the current value of a needed schedule-store attribute, then the GET service shall be used. An agent may also send scan event reports providing the manager with updates of the current attribute values, but this is not a mandated agent behavior.

The manager may request the schedule-store object attributes of the agent in which case the manager shall send the “Remote Operation Invoke | Get” command (see roiv-cmip-get in A.10.2 of ISO/IEEE 11073-20601:2016) with the handle value of the schedule-store object, as defined in the agent’s configuration. The agent shall respond by reporting its schedule-store object attributes to the manager using the “Remote Operation Response | Get” response (see rors-cmip-get in A.10.2 in ISO/IEEE 11073-20601:2016(. See H.1.1.2.2.1.1 for a full explanation of the GET operation for schedule-store metric data transmission.

E.1.6.2 SET service

There are currently no uses of the schedule-store SET service defined in this standard.

E.2 Schedule-segment class

E.2.1 General

An instance of the schedule-segment represents stored settings that define a series of scheduled entries. A schedule-segment object is not part of the static agent configuration because the number of instantiated schedule-segment instances may dynamically change. The manager accesses schedule-segment objects indirectly by methods and events of the schedule-store object.

E.2.2 Schedule-segment identification

The nomenclature code to identify the schedule-segment class is MDC_MOC_SCHEDULE_SEG.

E.2.3 Schedule-segment attributes

Table E.4 defines the set of schedule-segment attributes that are supported for PHD communication.

NOTE—The attribute qualifiers “static,” “dynamic,” and “observational” are omitted from Table E.4 since schedule-segments are dynamic (the object itself may change during the lifetime of an association).

The Fixed-Schedule-Segment-Data attribute may hold a sizable amount of data, depending on the agent capabilities and the application. An agent may choose to restrict the maximum size of the Fixed-Schedule-Segment-Data attribute in a way that is aligned with the maximum transmission unit of the transport system. In order to support this type of behavior, a manager that supports Schedule-segments shall be able to support the transfer of Fixed-Schedule-Segment-Data.

If two or more schedule-segments have the same Schedule-Segment-Start-Abs-Time or Schedule-Segment-Start-BO-Time, then the agent shall choose one segment as the Active-Schedule-Segment.

If the Schedule-Segment-End-Abs-Time or Schedule-Segment-End-BO-Time of an active schedule has occurred, then the agent shall update the value of the Active-Schedule-Segment.

E.2.4 Schedule-segment object methods

There are currently no schedule-segment object methods defined in this standard.

E.2.5 Schedule-segment object events

There are currently no schedule-segment object events defined in this standard.

E.2.6 Other schedule-segment services

There are currently no uses of the schedule-segment SET or GET services defined in this standard.

Table E.4—Schedule-segment attributes

Attribute name	Attribute ID	Attribute type	Remark	Qualifiers
Schedule-Segment-Instance-Number	MDC_ATTR_SCHED_SEG_INSTNO	InstNumber	The Instance-Number is the ID of a specific schedule-segment object instance. Each instance shall have a unique number assigned by the agent starting at 1. It is used by the manager to address a schedule-segment. Note, the Instance-Number value of 0 shall be reserved for reporting no active schedule-segment in the schedule-store.	Mandatory Static
Schedule-Segment-Entry-Map	MDC_ATTR_SCHED_SEG_MAP	ScheduleSegmentEntryMap	This attribute defines the format and contents of one schedule entry. An entry has a conditional header containing information applicable to all elements in the entry. If the schedule-segment period, start time, and entry interval are used, then the header information shall not be used. The entry then contains one or more elements, defined by the handle and an attribute value map defining the object attributes for each element in the schedule-segment.	Mandatory Dynamic
Schedule-Segment-Period	MDC_ATTR_SCHED_SEG_PERIOD	HighResRelativeTime	This attribute specifies the period of the schedule-segments. If the schedule-segment occurs only once, then the value shall be 0.	Mandatory Dynamic
Schedule-Segment-Entry-Interval	MDC_ATTR_SCHED_SEG_ENTRY_INTERVAL	HighResRelativeTime	This attribute specifies the interval between each entry in schedule-segments.	Conditional Dynamic
Schedule-Segment-Person-Id	MDC_ATTR_SCHED_SEG_PERSON_ID	PersonId	This standard supports devices that have simple support for data from multiple persons. A person ID is used to differentiate different persons. If the schedule-segment is able to have schedule data for multiple persons, it shall set the schedsc-multi-person bit in the Sched-Store-Capab attribute. If this bit is set, all schedule-segment instances shall support the Schedule-Segment-Person-Id attribute. Otherwise, this attribute is not defined.	Conditional Dynamic
Schedule-Segment-Entry-Count	MDC_ATTR_SCHED_SEG_ENTRY_CNT	INT-U32	This attribute gives the actual number of schedule-segment entries.	Optional Dynamic
Schedule-Segment-Label	MDC_ATTR_SCHED_SEG_LABEL_STRING	OCTET STRING	This attribute is an application-dependent label in printable ASCII for the schedule-segment to indicate its intended use and may be used for display purposes.	Optional Static
Schedule-Segment-LastUpdated-Abs-Time	MDC_ATTR_SCHED_SEG_LAST_UPDATED_ABS_TIME	AbsoluteTime	This attribute defines the time the schedule-segment was last updated. If this attribute is used, neither the Schedule-Segment-LastUpdated-HiRes-Time nor Schedule-Segment-LastUpdated-BO-Time shall be used.	Conditional Dynamic

Table E.4—Schedule-segment attributes *(continued)*

Attribute name	Attribute ID	Attribute type	Remark	Qualifiers
Schedule-Segment-LastUpdated-HiRes-Time	MDC_ATTR_SCHED_SEG_LAST_UPDATED_HIRES_TIME	HiResRelativeTime	This attribute defines the time the schedule-segment was last updated. If this attribute is used, neither the Schedule-Segment-LastUpdated-Abs-Time nor Schedule-Segment-LastUpdated-BO-Time shall be used.	Conditional Dynamic
Schedule-Segment-LastUpdated-BO-Time	MDC_ATTR_SCHED_SEG_LAST_UPDATED_BO_TIME	BaseOffsetTime	This attribute defines the time the schedule-segment was last updated. If this attribute is used, neither the Schedule-Segment-LastUpdated-Abs-Time nor Schedule-Segment-LastUpdated-HiRes-Time shall be used.	Conditional Dynamic
Schedule-Segment-Reference-Abs-Time	MDC_ATTR_SCHED_SEG_REF_ABS_TIME	AbsoluteTime	This attribute defines the time from which all entries within the schedule-segment are referenced. This attribute shall be used, if the Schedule-Segment-Reference-BO-Time is not used.	Conditional Dynamic
Schedule-Segment-Reference-BO-Time	MDC_ATTR_SCHED_SEG_REF_BO_TIME	BaseOffsetTime	This attribute defines the time from which all entries within the schedule-segment are referenced. This attribute shall be used, if the Schedule-Segment-Reference-Abs-Time is not used.	Conditional Dynamic
Schedule-Segment-Start-Abs-Time	MDC_ATTR_SCHED_SEG_START_ABS_TIME	AbsoluteTime	This attribute defines the time of the first execution of the schedule. If this attribute is used, the Schedule-Segment-Start-BO-Time shall not be used. If this attribute is used, the schedule-segment becomes the active-schedule-segment in the schedule-store at the start time. A schedule-segment shall not be active before the start time.	Optional Dynamic
Schedule-Segment-End-Abs-Time	MDC_ATTR_SCHED_SEG_END_ABS_TIME	AbsoluteTime	This attribute defines the expiration of the schedule. If this attribute is used, the Schedule-Segment-End-BO-Time shall not be used. If this attribute is used and the end time is reached, the schedule-segment shall no longer be the active-schedule-segment in the schedule-store.	Optional Dynamic
Schedule-Segment-Start-BO-Time	MDC_ATTR_SCHED_SEG_START_BO_TIME	BaseOffsetTime	This attribute defines the time of the first execution of the schedule. If this attribute is used, the Schedule-Segment-Start-Abs-Time shall not be used. If this attribute is used, the schedule-segment becomes the active-schedule-segment in the schedule-store at the start time. A schedule-segment shall not be active before the start time.	Optional Dynamic