IEEE STANDARDS ASSOCIATION

IEEE Standard for Fuzzy Markup Language

IEEE Computational Intelligence Society

Sponsored by the
Standards Committee

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

IEEE Std 1855™-2016

IEEE Standard for Fuzzy Markup Language

Sponsor

Standards Committee
of the
IEEE Computational Intelligence Society

Approved 29 January 2016

IEEE-SA Standards Board

Acknowledgments

Grateful acknowledgment is made to Springer Verlag for permission to reprint portions of the following source material:

"Fuzzy Markup Language: A XML Based Language for Enabling Full Interoperability in Fuzzy Systems Design," in *On the Power of Fuzzy Markup Language* [B1], © 2013.

"An Enhanced Visual Environment for Designing, Testing and Developing FML-based Fuzzy Systems," in *On the Power of Fuzzy Markup Language* [B2], © 2013.

"Distributing Fuzzy Reasoning through Fuzzy Markup Language: An Application to Ambient Intelligence," in *On the Power of Fuzzy Markup Language* [B3], © 2013.

"On the Need of a Standard Language for Designing Fuzzy Systems, in *On the Power of Fuzzy Markup Language* [B7], © 2013.

Abstract: A new specification language, named Fuzzy Markup Language (FML), is presented in this standard, exploiting the benefits offered by eXtensible Markup Language (XML) specifications and related tools in order to model a fuzzy logic system in a human-readable and hardware independent way. Therefore, designers of industrial fuzzy systems are provided with a unified and high-level methodology for describing interoperable fuzzy systems. The W3C XML Schema definition language is used by this standard to define the syntax and semantics of the FML programs.

Keywords: eXtensible markup language, FML, fuzzy logic, fuzzy logic controller, Fuzzy Markup Language, fuzzy system, hardware independence, IEEE 1855™, interoperability, labeled tree, XML, XML Schema

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

PDF: ISBN 978-1-5044-0677-2 STD20790
Print: ISBN 978-1-5044-0678-9 STDPD20790

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE documents are made available for use subject to important notices and legal disclaimers. These notices and disclaimers, or a reference to this page, appear in all standards and may be found under the heading "Important Notice" or "Important Notices and Disclaimers Concerning IEEE Standards Documents."

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

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

IEEE does not warrant or represent the accuracy or content of the material contained in its standards, and expressly disclaims all warranties (express, implied and statutory) not included in this or any other document relating to the standard, including, but not limited to, the warranties of: merchantability; fitness for a particular purpose; non-infringement; and quality, accuracy, effectiveness, currency, or completeness of material. In addition, IEEE disclaims any and all conditions relating to: results; and workmanlike effort. IEEE standards documents are supplied "AS IS" and "WITH ALL FAULTS."

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

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

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

Translations

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

Official statements

A statement, written or oral, that is not processed in accordance with the IEEE-SA Standards Board Operations Manual shall not be considered or inferred to be the official position of IEEE or any of its committees and shall not be considered to be, or be relied upon as, a formal position of IEEE. At lectures, symposia, seminars, or educational courses, an individual presenting information on IEEE standards shall make it clear that his or her views should be considered the personal views of that individual rather than the formal position of IEEE.

Comments on standards

Comments for revision of IEEE Standards documents are welcome from any interested party, regardless of membership affiliation with IEEE. However, IEEE does not provide consulting information or advice pertaining to IEEE Standards documents. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Since IEEE standards represent a consensus of concerned interests, it is important that any responses to comments and questions also receive the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to comments or questions except in those cases where the matter has previously been addressed. For the same reason, IEEE does not respond to interpretation requests. Any person who would like to participate in revisions to an IEEE standard is welcome to join the relevant IEEE working group.

Comments on standards should be submitted to the following address:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
Piscataway, NJ 08854 USA

Laws and regulations

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

Copyrights

IEEE draft and approved standards are copyrighted by IEEE under U.S. and international copyright laws. They are made available by IEEE and are adopted for a wide variety of both public and private uses. These include both use, by reference, in laws and regulations, and use in private self-regulation, standardization, and the promotion of engineering practices and methods. By making these documents available for use and adoption by public authorities and private users, IEEE does not waive any rights in copyright to the documents.

Photocopies

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

Updating of IEEE Standards documents

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

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

In order to determine whether a given document is the current edition and whether it has been amended through the issuance of amendments, corrigenda, or errata, visit the IEEE-SA Website at http://ieeexplore.ieee.org/xpl/standards.jsp or contact IEEE at the address listed previously. For more information about the IEEE-SA or IEEE's standards development process, visit the IEEE-SA Website at http://standards.ieee.org.

Errata

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

Patents

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

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

Participants

At the time this standard was completed, the IEEE P1855 Working Group had the following membership:

Giovanni Acampora, *Chair*
Bruno Di Stefano, *Vice Chair*

Plamen Angelov
Sansanee Auephanwiriyakul
Walter Banks
Aysenur Bilgin
Sergio Guadarrama
Kiyota Hashimoto
Uzay Kaymak
Vladik Kreinovich
Chang-Shing Lee
Chin-Teng Lin
Trevor Martin
Francisco José Moreno Velo
Witold Pedrycz
Marek Reformat
Autilia Vitiello
Dongrui Wu

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

Giovanni Acampora
Plamen Angelov
Kofi Appiah
Angelo Cangelosi
S. Claassen
Alessandro Di Nuovo
Bruno Di Stefano
Juan Manuel Escano
Randall Groves
Christian Guenther
Kiyota Hashimoto
Noriyuki Ikeuchi
Hisao Ishibuchi
Uzay Kaymak
Vladik Kreinovich
Chang-Shing Lee
Chin Teng Lin
Pedro Machado
Alyxander May
Thomas McGinnity
Mehran Panahi Akhavan
Evtim Peytchev
Marek Reformat
Thomas Starai
Wil van der Aalst
Eric Verbeek
Autilia Vitiello
Daidi Zhong

When the IEEE-SA Standards Board approved this standard on 29 January 2016, it had the following membership:

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

Chuck Adams
Masayuki Ariyoshi
Ted Burse
Stephen Dukes
Jianbin Fan
J. Travis Griffith
Gary Hoffman
Ronald W. Hotchkiss
Michael Janezic
Joseph L. Koepfinger*
Hung Ling
Kevin Lu
Annette D. Reilly
Gary Robinson
Mehmet Ulema
Yingli Wen
Howard Wolfman
Don Wright
Yu Yuan
Daidi Zhong

*Member Emeritus

Introduction

This introduction is not part of IEEE Std 1855-2016, IEEE Standard for Fuzzy Markup Language.

Currently, engineers and scientists developing a fuzzy system progress, iteratively, by alternating design, modeling, and simulation of their system. Depending on several design parameters, parts of this process are better handled by some software tools, while some others are better handled by other software tools. As a consequence and unfortunately, designers often have to enter various versions of their model into several software packages manually, slowing down the entire design phase of the system. The Fuzzy Logic (FL) Standard IEC 1131-7 was meant to address this issue, allow interoperability of various software packages, and allow engineers and scientists to use a single, well-defined specification language of fuzzy systems during all design activities. This has not happened mainly because of the lack of power of the abstractions of IEC 1131-7. In order to address this need, this standard defines an eXtensible Markup Language (XML)-based language, named Fuzzy Markup Language (FML), aimed at representing Fuzzy Logic Systems (FLSs) in an interoperable way. Considering the benefits offered by XML specifications and related tools, FML successfully allows FLSs to be implementable on different hardware platforms with a minimal effort, without additional design and implementation steps. Since FML was introduced by Giovanni Acampora, this markup language is contributing to the development and evolution of fuzzy-based intelligent decision systems belonging to different application fields such as healthcare, computer games, malware behavioral analysis, and many more [B4], [B13], [B22]. This standard uses the W3C XML Schema definition language to define the syntax and semantics of FML programs. In order to ensure wide acceptance throughout the user community, the XML Schema has been designed to encompass the most known FLSs: Mamdani [B16], Tsukamoto [B21], Takagi-Sugeno-Kang [B20], and AnYa [B5]. However, to accommodate all design needs, the language can be easily extended. This standard enables the generation of so-called FML drivers, i.e., software systems, based on XML technologies such as eXtensible Stylesheet Language Transformation (XSLT) and Document Object Model (DOM), capable of translating a system description based on FML to a runnable version.

This standardization project provides designers of industrial controllers and intelligent decision-making systems with a unified and high-level methodology for describing systems' behaviors by means of rules based on FL.

Contents

IEEE Standard for Fuzzy Markup Language

1. Overview

1.1 Scope

This standard defines an eXtensible Markup Language (XML)-based language, named Fuzzy Markup Language (FML), aimed at providing a unified and well-defined representation of Fuzzy Logic Systems (FLSs). This standard includes an extendable schema that natively defines the basic components of an FLS and enables the modeling of different categories of fuzzy inference engines, including Mamdani [B16], Tsukamoto [B21], Takagi-Sugeno-Kang (TSK) [B20], and AnYa [B5].

1.2 Purpose

The purpose of this standard is to allow the creation of interoperable FLSs. This standard uses the W3C XML Schema definition (XSD) language as the encoder, which allows for interoperability and the exchange of XML-based FLS instances between various systems. Different from other approaches used to describe fuzzy systems such as Fuzzy Control Language (FCL), FML allows fuzzy designers to simply code their ideas on heterogeneous hardware without need for a deep understanding of details related to the different platforms. This approach enables fuzzy systems designers to achieve *design transparency*. It means that, by using FML, it is possible to implement the same FLS on different hardware architectures with minimal effort and without additional design and implementation steps. In short, FML makes it possible to model an FLS in a human-readable and hardware-independent way.

1.3 The importance of the design of interoperable programmable controllers

The quality of design and maintenance steps of a generic industrial product strongly depends on the product documentation, e.g., written specifications, schematic diagrams, mechanical assembly drawings, and so on [B7]. If the documentation is correct and detailed, it is possible to transfer manufacturing of a product from one company to another just by transferring a complete and correct set of the applicable documentation [B7]. Different industrial standards define the required documentation for products belonging to different technological and industrial domains [B7]. For complex systems, there are families of standards encompassing all aspects of a product [B7]. For instance, the IEC 61131 family of standards on "Programmable Controllers" consists, at the time of this writing, of eight standards documents [B7]:

- Part 1: General information
- Part 2: Equipment requirements and tests
- Part 3: Programming languages
- Part 4: User guidelines
- Part 5: Messaging service specification
- Part 6: Communications via fieldbus (awaiting completion of fieldbus standards)
- Part 7: Fuzzy control programming
- Part 8: Guidelines for the application and implementation of programming languages

Part 7 is extremely important as it covers actual algorithms that power the programmable controllers [B7]. The nature of fuzzy logic (FL) is such that, without good and standardized documentation, it is impossible to understand how the control algorithm works [B7]. Part 7, Fuzzy control programming, has the explicit objective stated in the document: to provide the designers of fuzzy systems with a unified and common understanding of the basic means to integrate fuzzy reasoning in the Programmable Controller languages according to IEC 61131-3, as well as the possibility to exchange interoperable fuzzy control programs among different hardware architecture [B7]. The first goal, i.e., the one related to the programming languages, is probably less relevant today than the standards committee thought [B7]. In fact, IEC 61131-3 defines three graphical and two textual programming language standards for PLC (i.e., Programmable Logic Controller) [B7]:

- Ladder diagram (LD), graphical
- Function block diagram (FBD), graphical
- Structured text (ST), textual
- Instruction list (IL), textual
- Sequential function chart (SFC), graphical

This is clearly restrictive and reflects both the date of publication of the standard, December 1993, and the fact that it had been under revision for quite some time. Today, the object-oriented paradigm would probably be added [B7]. The second goal, i.e., the possibility to exchange interoperable fuzzy control programs among different hardware architectures, has become very relevant to practitioners and vendors of software for fuzzy systems, not only in the area of fuzzy control, but also in the more general area of FL applications [B7]. In many applications, fuzzy systems have useful benefits when compared to traditional controllers (see Annex A). However, the actual design of fuzzy systems heavily relies on hardware architectures, which will implement the running version of planned controllers. Indeed, even though fuzzy systems can be developed by using hardware solutions characterized by low costs and low complexity, their implementation is strongly bounded by the electrical, electronic, and programming constraints related to the particular hardware platform [B1]. The negative impacts of this dependence increase further when

fuzzy control approaches are used for modeling the behavior of ubiquitous frameworks in distributed computing environments composed by collections of interacting and heterogeneous hardware devices [B2].

The aim of FML is to bridge the aforementioned implementation gaps by introducing an abstract and unified approach for designing fuzzy systems in a hardware independent way [B1]. FML is a novel specific-purpose computer language that defines a detailed structure of fuzzy control independent of its legacy representation and improves systems designers' capabilities by providing them with a collection of features speeding up the whole development process of a centralized or distributed fuzzy system [B1].

1.4 Conventions used in this document

The conventions used throughout the document are included in this subclause. FML schema is case-sensitive. The naming convention for elements and attributes in the FML schema is the lower camel case. In detail, when several words are joined together, this naming convention sets the first letter of the entire word in lowercase, whereas, it puts subsequent first letters in uppercase. Examples of words in lower camel case are lowerCamelCase and thisIsAnExample.

1.4.1 Use of color in this standard

This standard uses a minimal amount of color to enhance readability. The coloring is not essential and does not affect the accuracy of this standard when viewed in black and white.

In formal language definitions, the used convention is as follows:

— **Boldface-blue** shows XML elements.

— **Boldface-red** shows XML attributes.

In example code, the used convention is as follows:

— `Blue` shows XML tags.

— `Red` shows XML attributes.

— **`Boldface-Green`** shows XML strings.

— `Black` shows the remaining XML components.

1.4.2 Diagram

The diagrams used throughout this standard graphically detail the organization of the elements and attributes in the FML schema definition. The meaning of the icons used in the diagrams is described in [B8].

1.4.3 Notational conventions

The keywords *required*, *shall*, *shall not*, *should*, *should not*, *recommended*, *may*, and *optional* in this document are to be interpreted as described in the IETF Best Current Practices document 14, RFC 2119 [B17].

1.4.4 Syntax examples

Any syntax example shown in this standard is for information only and solely intended to illustrate the use of such syntax.

1.5 Contents of this standard

The organization of the remainder of this standard is as follows:

- Clause 2 provides references to other applicable standards that are assumed or required for this standard.
- Clause 3 defines terms, acronyms, and abbreviations used throughout the different specifications contained in this standard.
- Clause 4 describes the fuzzy system model used to define FML.
- Clause 5 describes the schema structure.
- Clause 6 describes FML synthesis.
- Clause 7 describes the compatibility with and differences from the FCL.
- Clause 8 describes conformance rules.
- Clause 9 describes extensibility rules.

2. Normative references

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

W3C Recommendation (4 February 2004), Namespaces in XML 1.1.[1]

W3C Recommendation (28 October 2004), XML Schema Part 1: Structures, Second Edition.

W3C Recommendation (28 October 2004), XML Schema Part 2: Datatypes, Second Edition.

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.[2]

accumulation: A fuzzy operator that combines the output results of the rules belonging to a rule base in a final result.

activation: A fuzzy operator that implements the process by which the degree of fulfilment of the antecedent part of a fuzzy rule acts on the consequent part. *Syn:* **implication**.

aggregated fuzzy variable: A fuzzy variable whose linguistic terms are computed by means of fuzzy expressions involving linguistic terms of other fuzzy variables previously defined.

[1] W3C publications are available from the World Wide Web Consortium (https://www.w3.org/).
[2] *IEEE Standards Dictionary Online* is available at: http://ieeexplore.ieee.org/xpls/dictionary.jsp.

aggregation: A fuzzy operator that computes the degree of fulfilment of the antecedent part of a fuzzy rule.

antecedent: A fuzzy expression forming the IF-part of a fuzzy rule.

circular definition: A definition that specifies a linguistic term associated to a linguistic variable by means of a fuzzy expression containing linguistic terms previously defined on the same linguistic variable.

clause: A statement involving a linguistic variable, *V,* and one of its linguistic terms, *T,* in the form "*V* is *T*".

consequent: A fuzzy expression forming the THEN[-ELSE] part of a fuzzy rule.

crisp set: The classical bivalent set, in which the membership function only takes two values, commonly defined as 0 and 1.

crisp value: A numerical value.

defuzzification: Conversion of a fuzzy set into a crisp value.

degree of membership: A value ranging from 0 (not an element of the fuzzy set) to 1 (a member of the fuzzy set) that quantifies the membership of an element to a fuzzy set. *Syn:* **grade of membership**.

fuzzification: Conversion of a crisp value into a grade of membership for a linguistic term of a linguistic variable.

fuzzy expression: A *n*-ary relation from *n* fuzzy sets to [0,1]. Truth values, 0 and 1, and variables are fuzzy expressions. Fuzzy expressions combined with the logical operators *and*, *or,* and *not* are also fuzzy expressions.

fuzzy inference: The process that uses fuzzy logic to map a given input to an output.

fuzzy logic (FL): A mathematical logic in which statements do not have to be fully true or false.

fuzzy operator: A mapping from fuzzy sets to [0,1].

fuzzy rule: A fuzzy expression in the form of IF-THEN[-ELSE] involving linguistic variables and fuzzy operators.

fuzzy set: A mathematical set whose elements have degrees of membership, i.e., with the property that an element can be a member of the set, not a member of the set, or any of continuum of states of being a partial member of the set.

fuzzy system: A type of system in which the input/output mapping is based on fuzzy logic.

fuzzy term: *See:* **linguistic term**.

fuzzy variable: *See:* **linguistic variable**.

grade of membership: *See:* **degree of membership**.

hedge: *See:* **modifier**.

implication: *See:* **activation**.

linguistic term: A text label with an associated fuzzy set. *Syn:* **fuzzy term**.

linguistic variable: A variable that takes linguistic terms as values. *Syn:* **fuzzy variable**.

membership function: A function that expresses to which degree an element of a set belongs for a given linguistic term.

modifier: A process that turns a fuzzy set into a modified fuzzy set. *Syn:* **hedge**.

rule base: A collection of fuzzy rules to model a system's behavior.

rule weight: A value between 0 and 1 that states the degree of importance of a fuzzy rule.

singleton: A fuzzy set whose membership function is equal to one and only one at one point and equal to zero at all other points in the universe of discourse.

universe of discourse: The domain set of a membership function.

3.2 Acronyms and abbreviations

COA	center of area
COG	center of gravity
FCL	Fuzzy Control Language
FL	Fuzzy Logic
FLS	Fuzzy Logic System
FML	Fuzzy Markup language
MOM	mean of maxima
TSK	Takagi-Sugeno-Kang
WA	weighted average
XML	eXtensible Markup Language
XSD	XML Schema Definition
XSLT	eXtensible Stylesheet Language Transformation

4. A fuzzy system model for enabling a standard FML-based representation

The main purpose of this standard is to allow the creation of FLSs characterized by hardware interoperability. For this purpose, FML uses an alternative representation of an FLS (see Annex A) based on the notion of the labeled trees, which are data structures defined by means of the graph theory. As described in 4.1, it is possible to define an FLS through a labeled tree structure composed of two sets, Σ_{FLS} and E_{FLS}, which represent the collection of nodes modeling the main components of an FLS and the collection of parent-child relationships existing among them, respectively [B2]. Hereafter, a detailed description of an FLS labeled tree (as reported in [B2]) and its benefits is given.

4.1 FLS labeled tree—A novel vision of a fuzzy system

A labeled tree is a connected and acyclic graph where each vertex or edge, or both, is associated with a label. Formally, let Σ_V, Σ_E be finite alphabets of vertex labels and edge labels, respectively. Let V be a

finite nonempty set of vertices, *l* be a total function $l : V \to \Sigma_V$, *E* be a set of unordered distinct vertices called edges, and *a* be a total function $a : E \to \Sigma_E$. Then, G = (*V, l, E, a*) is a labeled graph. A labeled tree is a connected, acyclic labeled graph. The labels may be constants, node variables (corresponding to any node value), or path variables (corresponding to any path). Since the labeled trees are data models derived by the XML-based document representation, constants correspond to elements, attributes, and text values populating XML models. In detail, nodes labeled with text values are called *text nodes* and they are always *leaf nodes*. *Attribute nodes* can have only one child node, which is a text node. Also, any two attributes of a given element cannot have the same label. *Element nodes* can have zero or more child nodes that can be elements, attributes, or text nodes. Usually, the labels related to element nodes, attribute nodes, and text nodes are denoted with n_i, a_i, and t_i, respectively. Then, in the case of XML document modeling:

$$\Sigma_V = \bigcup_i \{n_i\} \cup \bigcup_i \{a_i\} \cup \bigcup_i \{t_i\}$$

$$\Sigma_E = \emptyset$$

In representing an FLS through labeled tree structures, it is necessary to individuate the different components of an FLS in order to map them to appropriate values for the labels n_i, a_i, and t_i, while at the same time, to establish the opportune parent-child relationships among different tree nodes. From a static point of view, an FLS contains a collection of fuzzy variables and fuzzy rules, which compose of the fuzzy knowledge base and fuzzy rule base, respectively. Hence, it is necessary to represent these two components as a labeled tree to obtain an FLS labeled tree.

To begin with, in the knowledge base, each fuzzy variable is defined mainly by means of a *name*, a *universe of discourse*, and a collection of fuzzy terms, where each fuzzy term can be represented through a pair (*Fuzzy Set*, *Linguistic Expression*). For example, consider an input fuzzy variable, named *var,* with three triangular fuzzy terms, named *term1*, *term2* and *term3*, and characterized by the measure unit *unit*. Figure 1 shows a graphical representation of the fuzzy variable *var*.

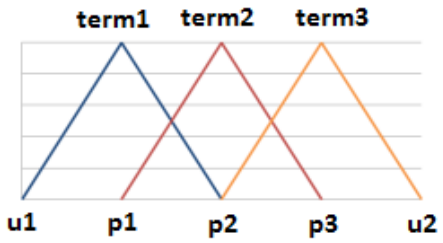


Figure 1—Fuzzy variable *var* [B2]

According to the description mentioned above, a set of tree labels featuring the fuzzy variable *var* can be defined as follows:

Element labels	Attribute labels	Text labels
$n_{var_1} = FuzzyVariable$	$a_{var_1} = a_{var_5} = a_{var_{10}} = a_{var_{15}} = name$	$t_{var_1} = var$
$n_{var_2} = n_{var_3} = n_{var4} = FuzzyTerm$	$a_{var_2} = domainLeft$	$t_{var_2} = u1$
$n_{var_5} = n_{var_6} = n_{var_7} = TriangularShape$	$a_{var_3} = domainRight$	$t_{var_3} = u2$
	$a_{var_4} = scale$	$t_{var4} = unit$
	$a_{var_6} = a_{var_{11}} = a_{var_{16}} = not$	$t_{var_5} = term1,\ t_{var_{10}} = term2,\ t_{var_{15}} = term3$
	$a_{var_7} = a_{var_{12}} = a_{var_{17}} = param1$	$t_{var_6} = t_{var_{11}} = t_{var_{16}} = false$
	$a_{var_8} = a_{var_{13}} = a_{var_{18}} = param2$	$t_{var_7} = p11,\ t_{var_{12}} = p21,\ t_{var_{17}} = p31$
	$a_{var_9} = a_{var_{14}} = a_{var_{19}} = param3$	$t_{var_8} = p1,\ t_{var_{13}} = p2,\ t_{var_{18}} = p3$
		$t_{var_9} = p31,\ t_{var_{14}} = p32,\ t_{var_{19}} = p33$

Let $\Sigma_{V_{var}}$ be the set of labels modeling the fuzzy variable *var*:

$$\Sigma_{V_{var}} = \{n_{var_1}, n_{var_2}, ..., n_{var_5}, a_{var_1}, a_{var_2}, ..., a_{var_9}, t_{var_1}, t_{var_2}, ..., t_{var_9}\}$$

and E_{var} be the set of parent-child relationships modeling the fuzzy variable *var*:

$$E_{var} = \{(n_{var_1}, n_{var_2}), ..., (n_{var_1}, n_{var_5}), (n_{var_1}, a_{var_1}), ..., (n_{var_1}, a_{var_4}), (n_{var_2}, a_{var_5}), (n_{var_2}, a_{var_5}), (n_{var_2}, a_{var_7}), (n_{var_5}, a_{var_7}), ..., (a_{var_1}, t_{var_1}), ..., (a_{var_9}, t_{var_9})\}$$

The graphical representation of the labeled tree for the fuzzy variable temperature is shown in Figure 2.

A more concise version of set $\Sigma_{V_{var}}$ and E_{var} could be obtained by considering subtrees representing the terms of the fuzzy variable. Considering a fuzzy variable made up by *m* terms, it is possible to define the sets $\Sigma_{V_{var}}$ and E_{var} as follows:

$$\Sigma_{V_{var}} = \bigcup_{i=1}^{m} \Sigma_{Vterm_i} \cup \Sigma_{V_{AFC}} \cup \{n_V\}$$

and

$$E_{var} = \bigcup_{i=1}^{m} E_{term_i} \cup E_{AFC}$$

where

$\Sigma_{V term_i}$ is the set of labels representing the labeled subtree of the *i*-th fuzzy term

$\Sigma_{V_{AFC}}$ is the set of labels for direct attributes of the fuzzy variable and their text

n_V is the root of tree representing the fuzzy variable

E_{term_i} is the set of parent-child relationships belonging to the labeled subtree of the *i*-th fuzzy term

E_{AFC} is the set of parent-child relationships that connect the fuzzy variable with its attributes and their text

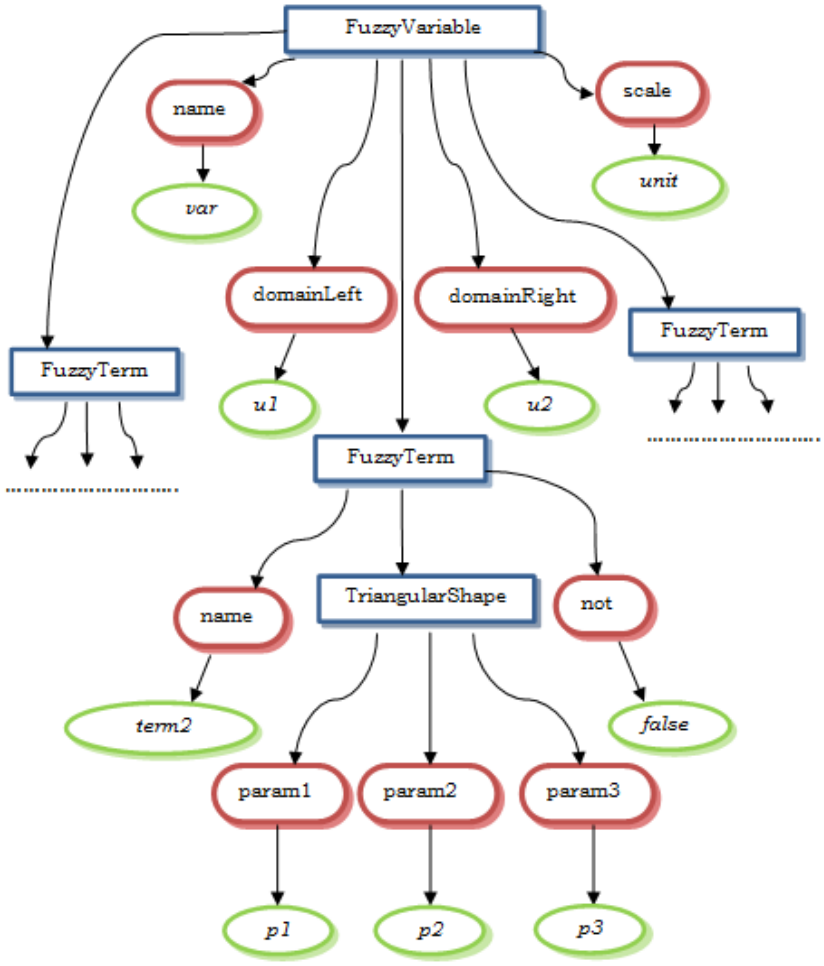


Figure 2—Labeled tree for the fuzzy variable *var*

By building a labeled tree for each one of k fuzzy variables FC in a knowledge base KB, it is possible to model a knowledge base through a labeled tree whose labels are as follows:

$$\Sigma_{KB} = \bigcup_i \Sigma_{V_{FC_i}} \cup \{n_{KB}\}$$

with the following parent-child relationships:

$$E_{KB} = \bigcup_i E_{V_{FC_i}} \cup \bigcup_i \{(n_{KB}, n_{FC_i})\}$$

where

$\Sigma_{V_{FC_i}}$, $E_{V_{FC_i}}$ are the label collection and the parent-child relationships modeling the i-th fuzzy variable, respectively

n_{KB} is a dummy root label joining the subtrees of the different fuzzy variables together

Similarly, it is possible to build the labeled tree modeling the fuzzy rule base, which is the second main component of an FLS, by defining the sets Σ_{RB} and E_{RB} as a rule base is composed of a set of fuzzy rules. The first step to build a labeled tree modeling a fuzzy rule base is to build a labeled tree modeling each individual rule. In a Mamdani fuzzy system, a fuzzy rule is formed by a fuzzy antecedent and a consequent

part and, recursively, the antecedents and consequents can be characterized by a sequence of fuzzy clauses, where, each fuzzy clause is represented by means of a pair (*FuzzyVariable*, *FuzzyTerm*) and possibly, a modifier. For example, let the first rule of a fuzzy rule base be as follows:

```
reg1: IF var1 is term1 opL var2 is term2 then var3 is mod1 term3
                                         else var4 is mod2 term4 (w)
```

The rule is characterized by the following components: 1) the antecedent part composed of two clauses, "*var1* is *term1"* and "*var2* is *term2"*, where *var1* and *var2* are fuzzy variables and *term1* and *term2* are fuzzy terms; 2) the consequent part composing of the THEN-part characterized by one clause "*var3* is *mod1 term3*" and the ELSE-part characterized by one clause "*var4* is *mod2 term4*", where *var3* and *var4* are fuzzy variables, *term3* and *term4* are fuzzy terms and *mod*1 and *mod2* are modifiers. The two clauses composing the antecedent part are connected through a connector *opL*, which is implemented through the method *and_M* if the connector *op* is the logical operator *and* or the method *or_M* if the connector *op* is the logical operator *or*. The value *w* enclosed into parentheses represents the weight of the rule. Accordingly, it is possible to identify the following labels (the subscript *fr* stands for fuzzy rule):

Element labels	Attribute labels	Text labels
$n_{fr_1} = Rule$	$a_{fr_1} = name$	$t_{fr_1} = reg1$
$n_{fr_2} = Antecedent$	$a_{fr_2} = weight$	$t_{fr_2} = w$
$n_{fr_3} = n_{fr_6} = n_{fr_{11}} = n_{fr_{15}} = Clause$	$a_{fr_3} = connector$	$t_{fr_3} = opL$
$n_{fr_4} = n_{fr_7} = n_{fr_{12}} = n_{fr_{16}} = Variable$	$a_{fr_4} = andMethod$	$t_{fr_4} = and\_M$
$n_{fr_5} = n_{fr_8} = n_{fr_{13}} = n_{fr_{17}} = Term$	$a_{fr_5} = orMethod$	$t_{fr_5} = or\_M$
$n_{fr_9} = Consequent$	$a_{fr_6} = modifier$	$t_{fr_6} = var1$
$n_{fr_{10}} = Then$	$a_{fr7} = modifier$	$t_{fr_7} = term1$
$n_{fr_{14}} = Else$		$t_{fr_8} = var2$
		$t_{fr_9} = term2$
		$t_{fr_{10}} = var3$
		$t_{fr_{11}} = term3$
		$t_{fr_{12}} = mod1$
		$t_{fr_{13}} = var4$
		$t_{fr_{14}} = term4$
		$t_{fr_{15}} = mod2$

The set composed of the described labels represents $\Sigma_{V_{fr}}$, whereas the set of parent-child relationships E_{fr} for the *i*-th rule is as follows:

$$E_{fr} = \{(n_{fr_1}, n_{fr_2}), (n_{fr_1}, n_{fr_9}), (n_{fr_1}, a_{fr_1}), ..., (n_{fr_1}, a_{fr_5}), (a_{fr_1}, t_{fr_1}), ..., (a_{fr_5}, t_{fr_5}), (n_{fr_2}, n_{fr_3}),$$
$$(n_{fr_2}, n_{fr_6}), ..., (n_{fr_4}, t_{fr_6}), ..., (n_{fr_5}, n_{fr_7}), ...\}$$

Figure 3 shows the labeled tree for the first rule of rule base considered in our example.

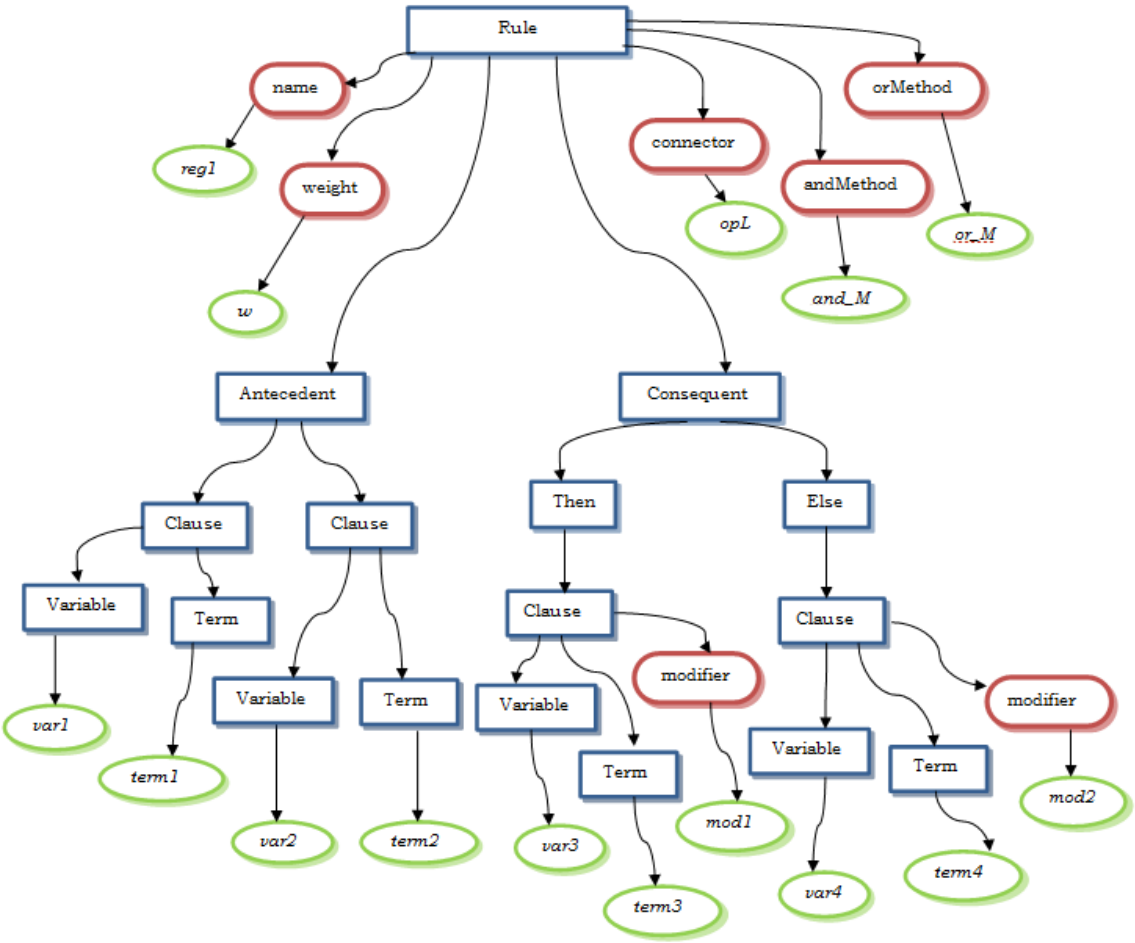


Figure 3—Labeled tree for the rule reg1

For each rule *i*, it is possible to build the sets $\Sigma_{V_{R_i}}$ and E_{R_i} by following the same procedure adopted for the first rule. By summarizing, it is possible to build the labeled tree modeling the fuzzy rule base defining the sets Σ_{RB} and E_{RB} in the following way:

$$\Sigma_{RB} = \bigcup_i \Sigma_{V_{R_i}} \cup \{n_{RB}\} \cup \Sigma_{V_{RA}}$$

$$E_{RB} = \bigcup_i E_{R_1} \cup \bigcup_i \{(n_{RB}, n_{R_{i1}})\} \cup E_{RA}$$

where

$\Sigma_{V_{R_i}}$, E_{R_i} are the collection of labels and the parent-child relationships modeling *i*-th fuzzy rule, respectively

n_{RB} is a dummy root label joining together the subtrees of the different fuzzy rules

$\Sigma_{V_{RA}}$ represents the set of labels related to the attributes of the rule base and their texts

E_{RA} represents the set of parent-child relationships existing between the node n_{RB} and the attributes of the rule base and between the attributes and their texts

Finally, a FLS can be modeled by means of a labeled tree defined by the following sets:

$$\Sigma_{FLS} = \Sigma_{KB} \cup \Sigma_{RB} \cup \{n_{FLS}\}$$

$$E_{FLC} = E_{KB} \cup E_{RB} \cup \{(n_{FLS}, n_{KB}), (n_{FLS}, n_{RB})\}$$

where

n_{FLS} is a dummy node joining the fuzzy rule base and the knowledge base

Figure 4 shows a labeled tree for a whole FLS. In this figure, it is possible to view also the sets $\Sigma_{V_{RA}}$ and E_{RA} of the rule base.

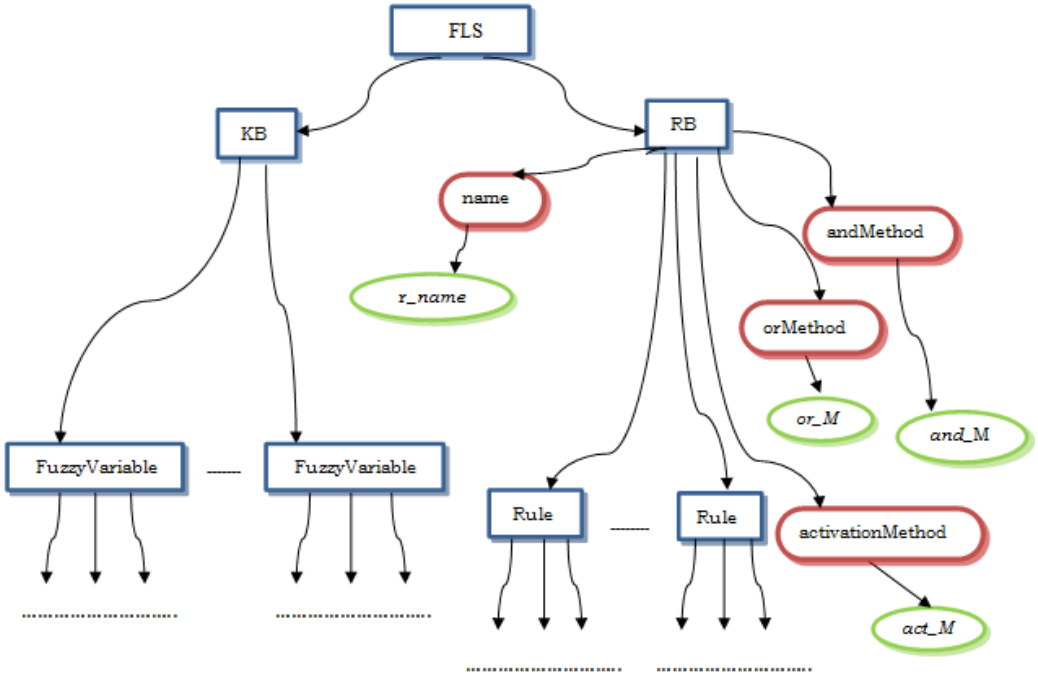


Figure 4—A partial FLS labeled tree

4.2 Benefits of FLS labeled trees

FML achieves its purpose by using the labeled tree appraoch for modeling FLSs. This novel structural vision of a fuzzy system makes it particularly simple to define a tag based grammar capable of hierarchically depicting different fuzzy components [B7]. Besides, the hierarchical nature of the labeled tree representation used by FML allows the designers to define their FLSs by using XML technologies and, consequently, to strongly improve the interoperability of the designed systems [B7]. Moreover, since XML is used for distributing information in complex distributed environments, FML description of a fuzzy system enables the distribution of fuzzy computation in ubiquitous environments as could be required from the current trends of computing [B7]. Finally, because labeled trees are characterized by a well-defined hierarchical shape, the designers of fuzzy systems can visually implement FML systems and, consequently, can reduce the time for designing a generic fuzzy framework [B7].

5. FML Schema definition

FML aims to effectively design and implement FLSs. FML programs are coded through a collection of correlated semantic tags capable of modeling the different components of a classical FLS by exploiting abstraction benefits offered by FLS labeled trees. Indeed, the XML document-like structure of FLS labeled trees allows an immediate definition of a new specification language for FLS based on XML tools. In particular, each element node in the FLS labeled tree will represent an XML tag, each attribute node will represent an XML attribute, each text node will represent a value for XML attributes, and, finally, the parent-child relation will represent the relations among XML elements and attributes (grammar syntax). Hereafter, the details about the FML Schema definition are given.

5.1 General information

All FML data elements defined in 5.2 shall be defined in the namespace http://www.ieee1855.org. This namespace is reserved by this standard and shall not be used to represent any other data element (see W3C Recommendation, Namespaces in XML 1.1). This namespace shall not be used to define extensions to an FML-based FLS instance. The namespace is used in the XSD provided with this standard (see Annex B). If an organization provides extensions to an FML-based FLS instance, the organization should define the namespace for these extensions.

5.2 FML schema structure

The schema of the FML specification is used to describe the basic components of FLSs. The element **fuzzySystem** is the top-level element of this schema. Such an element represents the root tag of FML programs, that is, the opening tag of each FML program.

5.2.1 Description

The top element **fuzzySystem** (see Figure 5) has two attributes defined as follows:

— The attribute **name** is used to specify a unique name for the FLS.

— The attribute **networkAddress** is used to define the location of the fuzzy system in a computer network.

Table 1 shows properties of the attributes of the element **fuzzySystem**.

The top-level element **fuzzySystem** contains one element **knowledgeBase** and one or multiple elements chosen from the elements **mamdaniRuleBase**, **tsukamotoRulBase**, **tskRuleBase**, and **anYaRuleBase**, as shown in Figure 5.

Table 1—Properties of the attributes of the element fuzzySystem

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0" [B6].	—
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	`127.0.0.1`

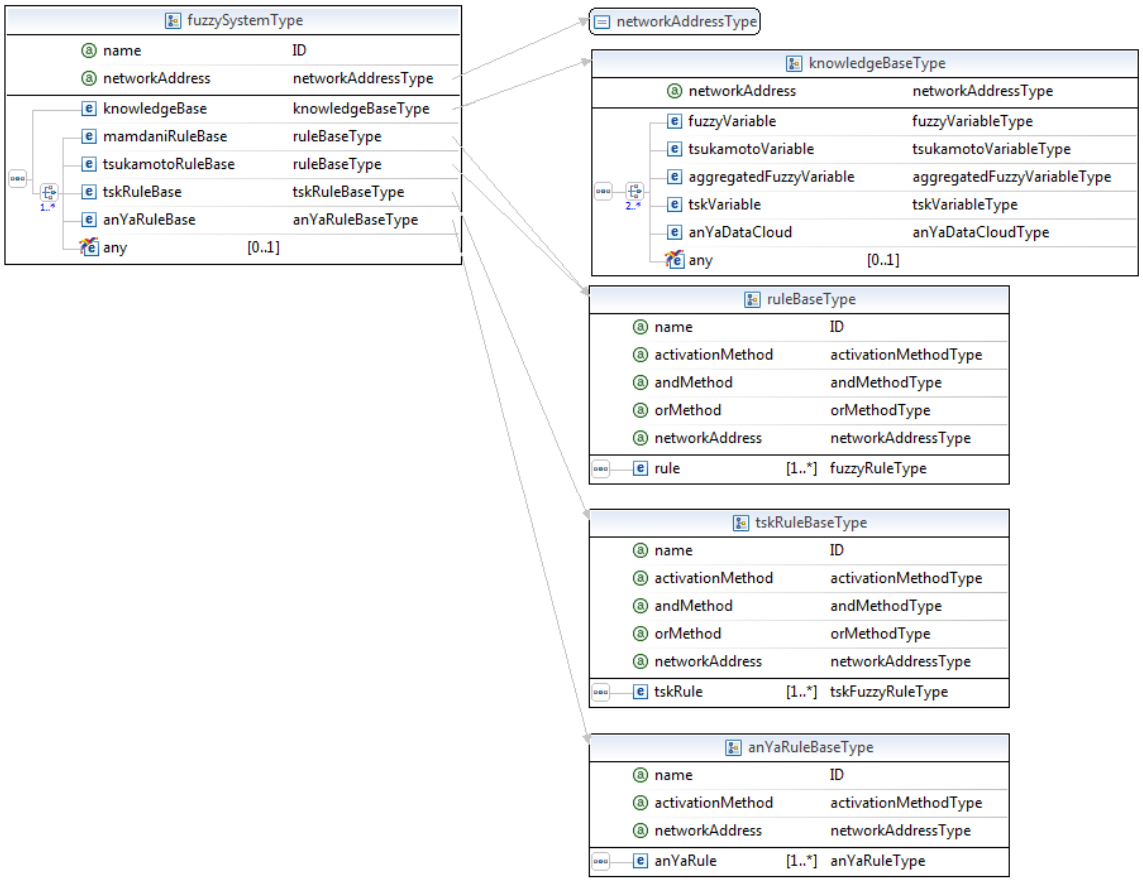


Figure 5—Type diagram for the elements fuzzySystem, knowledgeBase, mamdaniRuleBase, tsukamotoRuleBase, tskRuleBase, and anYaRuleBase

The element **knowledgeBase** (see Figure 5) represents the knowledge base that maintains the set of fuzzy variables used to model the fuzzy rule base. It has the attribute **networkAddress** that defines the location of the knowledge base in a computer network, which may be different from the rest of the parts of the fuzzy system.

The element **knowledgeBase** contains one or multiple elements chosen from the following elements (see Figure 5):

- The element **fuzzyVariable** represents a fuzzy variable related to a given FLS.
- The element **tsukamotoVariable** represents an output fuzzy variable that can be used in a rule of a Tsukamoto system.
- The element **aggregatedFuzzyVariable** represents an input fuzzy variable that can be used to aggregate two or more fuzzy variables related to a given FLS.
- The element **tskVariable** represents an output fuzzy variable that can be used in a rule of a TSK system.
- The element **anYaDataCloud** represents a data cloud that can be used in a rule antecedent part of a data cloud-based FLS.

The element **fuzzyVariable** (see Figure 6) has nine attributes, as follows:

- The attribute **name** is used to define a unique name for a fuzzy variable.
- The attribute **scale** is used to define the scale used to measure a fuzzy variable.
- The attribute **domainLeft** is used to model the left boundary of the universe of discourse of a fuzzy variable.
- The attribute **domainRight** is used to model the right boundary of the universe of discourse of a fuzzy variable.
- The attribute **type** is used to define the position of a fuzzy variable in a rule (consequent part or antecedent part).
- The attribute **accumulation** is used to define the method of accumulation, i.e., a method that permits the combination of results of an output fuzzy variable of each rule in a final result.
- The attribute **defuzzifier** is used to define the method used to execute the conversion from a fuzzy set, obtained after aggregation process, into a crisp value.
- The attribute **defaultValue** is used to define a real value used only when no rule has fired for the variable at issue.
- The attribute **networkAddress** is used to define the location of the fuzzy variable in a computer network.

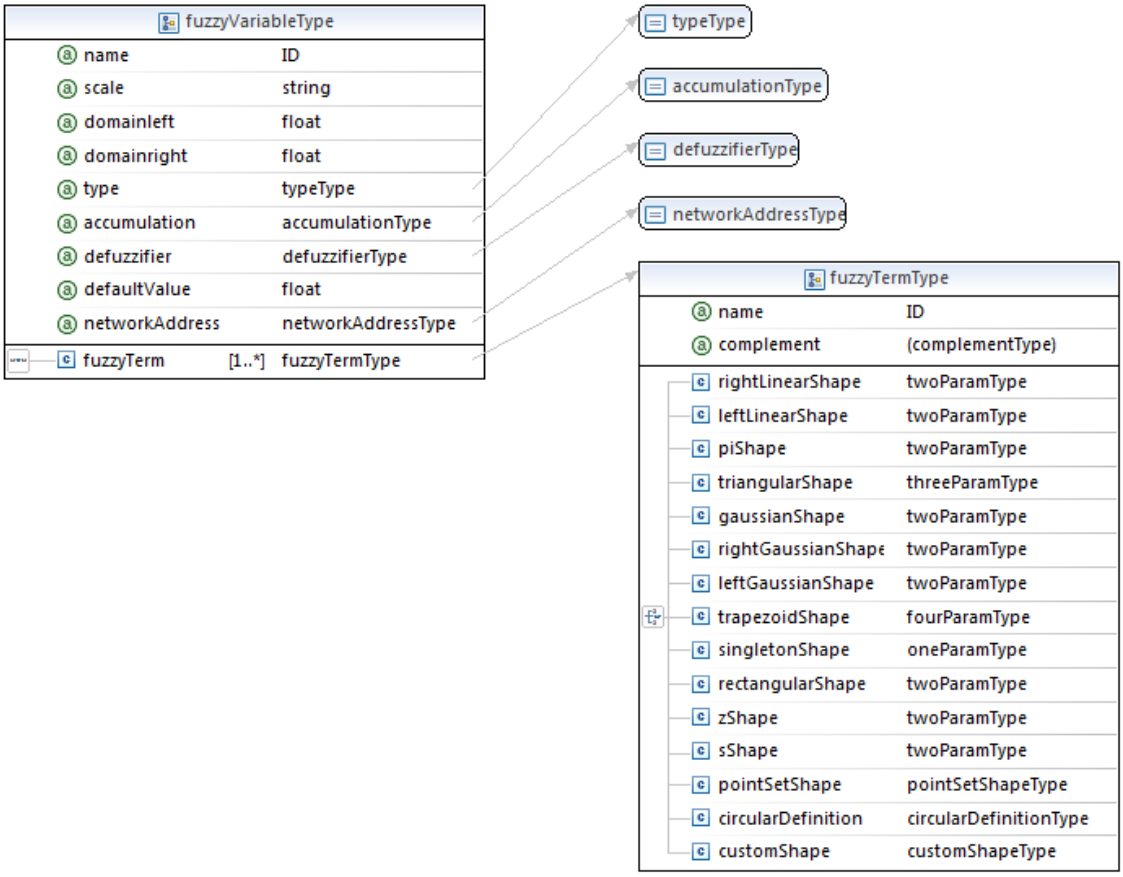


Figure 6—Type diagram for the element fuzzyVariable

Table 2 shows properties of the attributes of the element **fuzzyVariable**.

Table 2—Properties of the attributes of the element fuzzyVariable

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0" [B6].	—
scale	optional	Any string.	Not provided
domainLeft	required	Any float.	—
domainRigth	required	Any float.	—
type	optional	Any string that conforms to the following pattern: `(input|output|INPUT|OUTPUT|Input|Output)`	**`input`**

Table continues

Table 2—Properties of the attributes of the element fuzzyVariable *(continued)*

Name	Use	Values	Default value
accumulation	optional	Any string that conforms to the following patterns: – `MAX` for implementing the accumulation with the maximum as defined from Equation (A.35); – `PROBOR` for implementing the accumulation with the probabilistic sum as defined from Equation (A.36); – `BSUM` for implementing the accumulation with the bounded sum as defined from Equation (A.37); – `DRS` for implementing the accumulation with the drastic sum as defined from Equation (A.38); – `ESUM` for implementing the accumulation with the Einstein sum as defined from Equation (A.39); – `HSUM` for implementing the accumulation with the Hamacher sum as defined from Equation (A.40); – `NILMAX` for implementing the accumulation with the Nilpotent maximum as defined from Equation (A.41); – `custom_\S*` for a custom accumulation method. NOTE: Use of the last pattern results in a document *conformant* but not *strictly conformant*.	**`MAX`**
defuzzifier	optional	Any string that conforms to the following patterns: – `MOM` for the defuzzifier method named mean of maxima as defined from Equation (A.42); – `LM` for the defuzzifier method named leftmost maximum as defined from Equation (A.43); – `RM` for the defuzzifier method named rightmost maximum as defined from Equation (A.44); – `COG` for the defuzzifier method named center of gravity as defined from Equation (A.45); – `COA` for the defuzzifier method nmed center of area as defined from Equation (A.46); – `custom_\S*` for a custom defuzzifier method. NOTE: Use of the last pattern results in a document *conformant* but not *strictly conformant*.	**`COG`**
defaultValue	optional	Any float.	**`0`**
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	**`127.0.0.1`**

The element **fuzzyVariable** contains one or multiple elements **fuzzyTerm**, as shown in Figure 6.

The element **fuzzyTerm** (see Figure 7) defines a linguistic term describing the fuzzy variable. It has two attributes, as follows:

— The attribute **name** is used to define a unique name for the linguistic term.

— The attribute **complement** is a Boolean attribute and it is used to define if it is necessary to consider the complement of the membership function defined by given parameters.

Table 3 shows properties of the attributes of the element **fuzzyTerm**.

Table 3—Properties of the attributes of the element fuzzyTerm

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a NCName in the Recommendation "Namespaces in XML 1.0" [B6].	—
complement	optional	Any string that conforms to the following pattern: `(true|false|TRUE|FALSE|True|False).`	`false`

The element **fuzzyTerm** contains one element chosen from the following elements (see Figure 7):

— The element **triangularShape** specifies the triangular shape for the fuzzy term.

— The element **rightLinearShape** specifies the right linear shape for the fuzzy term.

— The element **leftLinearShape** specifies the left linear shape for the fuzzy term.

— The element **piShape** specifies the PI shape for the fuzzy term.

— The element **gaussianShape** specifies the gaussian shape for the fuzzy term.

— The element **rightGaussianShape** specifies the right gaussian shape for the fuzzy term.

— The element **leftGaussianShape** specifies the left gaussian shape for the fuzzy term.

— The element **trapezoidShape** specifies the trapezoidal shape for the fuzzy term.

— The element **sShape** specifies the S shape for the fuzzy term.

— The element **zShape** specifies the Z shape for the fuzzy term.

— The element **rectangularShape** specifies the rectangular shape for the fuzzy term.

— The element **singletonShape** specifies the singleton shape for the fuzzy term.

— The element **pointSetShape** specifies a shape for the fuzzy term by using a set of points defined by the user.

— The element **circularDefinition** specifies a fuzzy term by means of a logical expression containing fuzzy terms previously defined.

— The element **customShape** specifies a custom shape for the fuzzy term.

NOTE—Use of this element and all its nested elements results in a document *conformant* but not *strictly conformant*.

Every shaping element (**rightLinearShape**, **leftLinearShape**, **piShape**, **gaussianShape**, **rightGaussianShape**, **leftGaussianShape**, **rectangularShape**, **sShape**, **zShape, triangularShape**, **trapezoidShape**, **singletonShape**, and **userShape**) uses a set of attributes that defines the parameters required for the corresponding fuzzy set. The number of these attributes depends on the chosen fuzzy set shape.

In detail, the element **singletonShape** has an only attribute (see Figure 7), as follows:

— The attribute **param1** is used to define the first parameter characterizing the fuzzy set shape.

Table 4 shows properties of the attribute of the element **singletonShape**.

Table 4—Properties of the attributes of the element singletonShape

Name	Use	Values	Default value
param1	required	Any float.	—

The elements **rightLinearShape**, **leftLinearShape**, **piShape**, **gaussianShape**, **rightGaussianShape**, **leftGaussianShape**, **rectangularShape**, **sShape**, and **zShape** has two attributes (see Figure 7) as follows:

— The attribute **param1** is used to define the first parameter characterizing the fuzzy set shape.

— The attribute **param2** is used to define the second parameter characterizing the fuzzy set shape.

Table 5 shows properties of the attributes of the elements **rightLinearShape**, **leftLinearShape**, **piShape**, **gaussianShape**, **rightGaussianShape**, **leftGaussianShape**, **rectangularShape**, **sShape**, and **zShape**.

Table 5—Properties of the attributes of the elements rightLinearShape, leftLinearShape, piShape, gaussianShape, rightGaussianShape, leftGaussianShape, rectangularShape, sShape, and zShape

Name	Use	Values	Default value
param1	required	Any float.	—
param2	required	Any float.	—

The element **triangularShape** has three attributes (see Figure 7) as follows:

— The attribute **param1** is used to define the first parameter characterizing the fuzzy set shape.

— The attribute **param2** is used to define the second parameter characterizing the fuzzy set shape.

— The attribute **param3** is used to define the third parameter characterizing the fuzzy set shape.

Table 6 shows properties of the attributes of the element **triangularShape**.

Table 6—Properties of the attributes of the element triangularShape

Name	Use	Values	Default value
param1	required	Any float.	—
param2	required	Any float.	—
param3	required	Any float.	—

The element **trapezoidalShape** has four attributes (see Figure 7) as follows:

— The attribute **param1** is used to define the first parameter characterizing the fuzzy set shape.

— The attribute **param2** is used to define the second parameter characterizing the fuzzy set shape.

— The attribute **param3** is used to define the third parameter characterizing the fuzzy set shape.

— The attribute **param4** is used to define the fourth parameter characterizing the fuzzy set shape.

Table 7 shows properties of the attributes of the element **trapezoidalShape**.

Table 7—Properties of the attributes of the element trapezoidalShape

Name	Use	Values	Default value
param1	required	Any float.	—
param2	required	Any float.	—
param3	required	Any float.	—
param4	required	Any float.	—

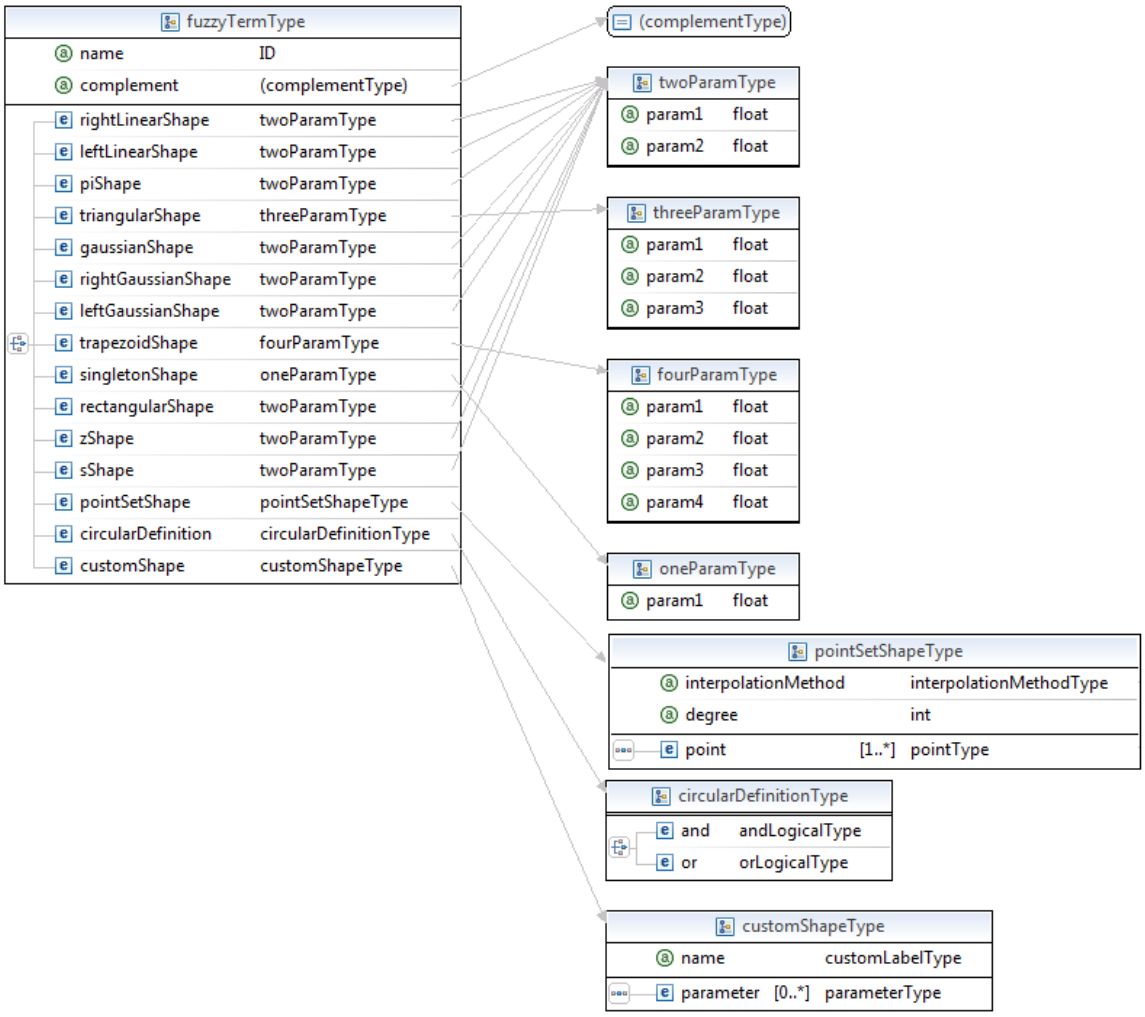


Figure 7—Type diagram for the elements fuzzyTerm, rightLinearShape, leftLinearShape, piShape, gaussianShape, rightGaussianShape, leftGaussianShape, rectangularShape, sShape, zShape, triangularShape, trapezoidShape, and singletonShape

The element **pointSetShape** has two attributes (see Figure 7), as follows:

— The attribute **interpolationMethod** is used to define the method to interpolate the points defined by the user.

— The attribute **degree** is used to define the polynomial degree to be employed during the interpolation process (if the interpolation method allows to define a degree).

Table 8 shows properties of the attribute of the element **pointSetShape**.

Table 8—Properties of the attributes of the element pointSetShape

Name	Use	Values	Default value
interpolationMethod	optional	Any string that conforms to the following patterns: – `linear` for implementing the Linear interpolation as defined in [B11]; – `lagrange` for implementing the Lagrange interpolation as defined in [B10]; – spline for implementing the Spline interpolation as defined in [B12].	`linear`
degree	optional	Any int.	`3`

The element **pointSetShape** contains two or multiple elements **point**. The element **point** represents a point of the geometric area defining the set point based fuzzy shape. It has two attributes (see Figure 8), as follows:

— The attribute **x** is used to define the coordinate on the x-axis.

— The attribute **y** is used to define the coordinate on the y-axis.

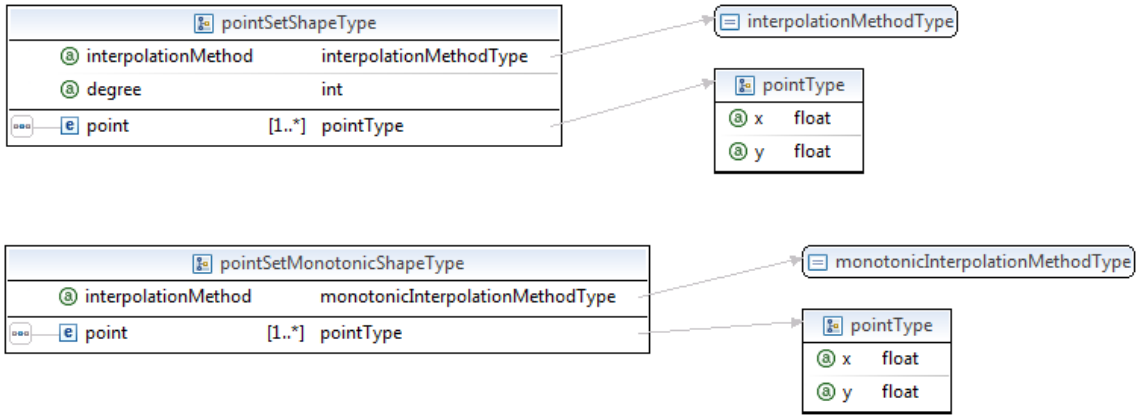


Figure 8—Type diagram for the element pointSetShape and pointSetMonotonicShape

Table 9 shows properties of the attributes of the element **point**.

Table 9—Properties of the attributes of the element point

Name	Use	Values	Default value
x	required	Any float.	—
y	required	Any float.	—

The element **circularDefinition** contains one element chosen from the following elements: **and**, **or** (see Figure 7). The element **and** has an only attribute (see Figure 9):

— The attribute **t-norm** is used to define the method to be associated to the FL operator *and*.

Table 10 shows properties of the attributes of the element **and**.

Table 10—Properties of the attributes of the element and

Name	Use	Values	Default value
t-norm	optional	Any string that conforms to the following patterns: – `MIN` for implementing the operator *and* with the minimum as defined from Equation (A.14); – `PROD` for implementing the operator *and* with the product as defined from from Equation (A.15); – `BDIF` for implementing the operator *and* with bounded difference as defined from from Equation (A.16); – `DRP` for implementing the operator *and* with the drastic product as defined from from Equation (A.17); – `EPROD` for implementing the operator *and* with the Einstein product as defined from from Equation (A.18); – `HPROD` for implementing the operator *and* with the Hamacher product as defined from from Equation (A.19); – `NILMIN` for implementing the operator *and* with the Nilpotent minimum as defined from from Equation (A.20); – `custom_\S*` for a custom method for operator *and*.	`MIN`
NOTE—Use of the last pattern results in a document *conformant* but not *strictly conformant*.			

At the same way, the element **or** has an only attribute (see Table 9):

— The attribute **t-conorm** is used to define the method to be associated to the FL operator *or*.

Table 11 shows properties of the attributes of the element **or**.

Table 11—Properties of the attributes of the element or

Name	Use	Values	Default value
t-conorm	optional	Any string that conforms to the following patterns: – `MAX` for implementing the connector *or* with the maximum as defined from from Equation (A.21); – `PROBOR` for implementing the connector *or* with the probabilistic sum as defined from from Equation (A.22); – `BSUM` for implementing the operator *or* with the bounded sum as defined from from Equation (A.23); – `DRS` for implementing the operator *or* with the drastic sum as defined from from Equation (A.24); – `ESUM` for implementing the operator or with the Einstein sum as defined from from Equation (A.25); – `HSUM` for implementing the operator *or* with the Hamacher sum as defined from from Equation (A.26); – `NILMAX` for implementing the operator *or* with the Nilpotent maximum as defined from from Equation (A.27); – `custom_\S*` for a custom method for operator *or*.	`MAX`
NOTE: Use of the last pattern results in a document *conformant* but not *strictly conformant*.			

Both the element **and** and the element **or** contain two elements, **termName** or an element **and** followed by an element **termName** or an element **or** followed by an element **termName** (see Figure 9). The element **termName** represents the term to be used for describing the term under definition. The element **termName** has an only attribute (see Figure 9), as follows:

— The attribute **complement** is used to point out if it is necessary to consider the complement of the membership function of the corresponding term.

Table 12 shows properties of the attributes of the element **termName**.

Table 12—Properties of the attributes of the element termName

Name	Use	Values	Default value
complement	optional	Any string that conforms to the following pattern: `(true|false|TRUE|FALSE|True|False).`	`false`

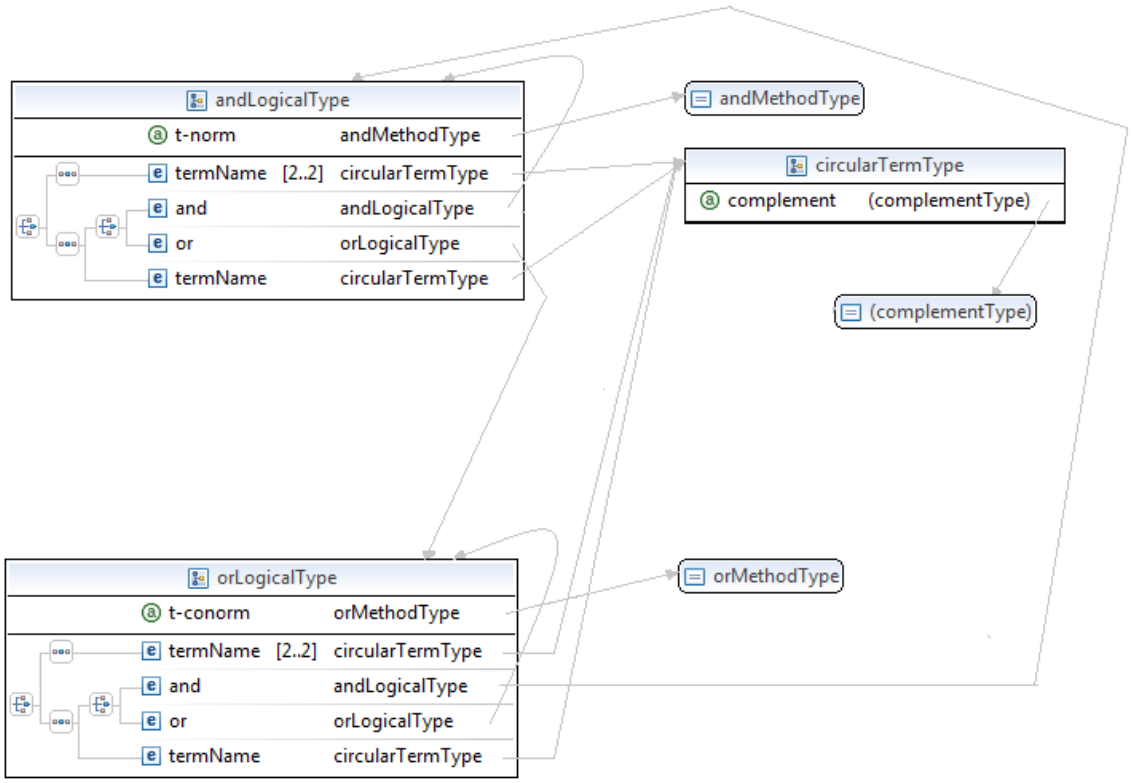


Figure 9—Type diagram for the elements and, or, and termName

The element **customShape** (see Figure 7) has an only attribute as follows:

— The attribute **name** is used to define a name for the custom shape.

Table 13 shows properties of the attributes of the element **customShape**.

Table 13—Properties of the attributes of the element customShape

Name	Use	Values	Default value
name	required	Any string that conforms to the following pattern: `(custom_\S*).`	—

The element **customShape** contains zero or more elements **parameter** that represents a parameter useful to define the custom shape for a fuzzy term. The element **parameter** (see Figure 10) has an only attribute as follows:

— The attribute **value** is used to define the value for a parameter characterizing the custom shape.

Table 14 shows properties of the attributes of the element **parameter**.

Table 14—Properties of the attributes of the element parameter

Name	Use	Values	Default value
value	required	Any string.	—

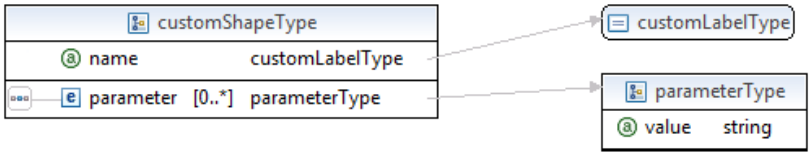


Figure 10—Type diagram for the elements customShape and parameter

The element **tsukamotoVariable** (see Figure 11) has eight attributes as follows:

— The attribute **name** is used to define a unique name for a fuzzy variable.

— The attribute **scale** is used to define the scale used to measure a fuzzy variable.

— The attribute **domainLeft** is used to model the left boundary of the universe of discourse of a fuzzy variable.

— The attribute **domainRight** is used to model the right boundary of the universe of discourse of a fuzzy variable.

— The attribute **type** is used to define the position of a fuzzy variable into rule (consequent part or antecedent part).

— The attribute **combination** is used to define the method of aggregation of the Tsukamoto rule outputs to obtain a final result.

— The attribute **defaultValue** is used to define a real value used only when no rule has fired for the variable at issue.

— The attribute **networkAddress** is used to define the location of the fuzzy variable in a computer network.

Table 15 shows properties of the attributes of the element **tsukamotoVariable**.

The element **tsukamotoVariable** contains one or multiple elements **tsukamotoTerm**, as shown in Figure 11.

Table 15—Properties of the attributes of the element tsukamotoVariable

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0" [B6].	—
scale	optional	Any string.	Not provided
domainLeft	required	Any float.	—
domainRigth	required	Any float.	—
type	required	The string `output`.	—
combination	optional	Any string that conforms to the following patterns: – `WA` for the weighted average as defined from Equation (A.47); – `custom_\S*` for a custom aggregation of the TSK rule outputs. NOTE: Use of the last pattern results in a document *conformant* but not *strictly conformant*.	**WA**
defaultValue	optional	Any float.	**0**
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	**127.0.0.1**

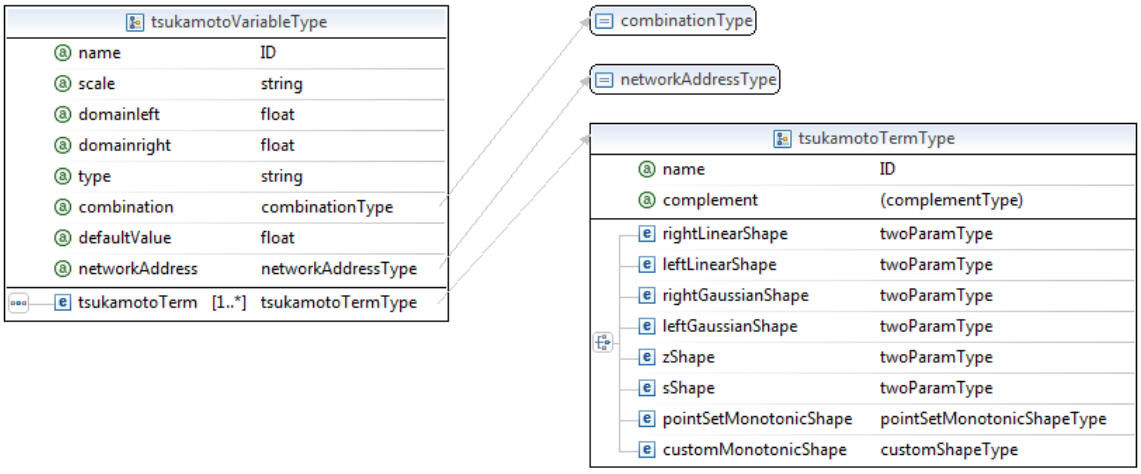


Figure 11—Type diagram for the elements tsukamotoVariable and tsukamotoTerm

The element **tsukamotoTerm** represents a monotonic membership function. It has two attributes (see Figure 11) as follows:

— The attribute **name** is used to define a unique name for the linguistic term.

— The attribute **complement** is a Boolean attribute and it is used to define if it is necessary to consider the complement of the membership function defined by given parameters.

Table 16 shows properties of the attributes of the element **tsukamotoTerm**.

Table 16—Properties of the attributes of the element tsukamotoTerm

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a NCName in the Recommendation "Namespaces in XML 1.0" [B6]	—
complement	optional	Any string that conforms to the following pattern: `(true|false|TRUE|FALSE|True|False).`	`false`

The element **tsukamotoTerm** contains one element chosen from the following elements (see Figure 11):

— The element **rightLinearShape** specifies the right linear shape for the fuzzy term.

— The element **leftLinearShape** specifies the left linear shape for the fuzzy term.

— The element **rightGaussianShape** specifies the right gaussian shape for the fuzzy term.

— The element **leftGaussianShape** specifies the left gaussian shape for the fuzzy term.

— The element **sShape** specifies the S shape for the fuzzy term.

— The element **zShape** specifies the Z shape for the fuzzy term.

— The element **pointSetMonotonicShape** specifies a monotonic shape for the fuzzy term based on a set of points defined by the user.

— The element **customMonotonicShape** specifies a custom monotonic shape for the fuzzy term.

NOTE—Use of this element and all its nested elements results in a document *conformant* but not *strictly conformant*.

Every shaping element (**rightLinearShape**, **leftLinearShape**, **rightGaussianShape**, **leftGaussianShape**, **sShape**, **zShape**, and **userMonotonicShape**) uses a set of attributes that defines the parameters required for the corresponding fuzzy set. The number of these attributes depends on the chosen fuzzy set shape.

The elements **rightLinearShape**, **leftLinearShape**, **rightGaussianShape**, **leftGaussianShape**, **sShape**, and **zShape** have two attributes (see Figure 7) as follows:

— The attribute **param1** is used to define the first parameter characterizing the fuzzy set shape.

— The attribute **param2** is used to define the second parameter characterizing the fuzzy set shape.

Table 5 shows properties of the attributes of the elements **rightLinearShape**, **leftLinearShape**, **rightGaussianShape**, **leftGaussianShape**, **sShape**, and **zShape**.

The element **pointSetMonotonicShape** has an only attribute (see Figure 8) as follows:

— The attribute **interpolationMethod** is used to define a method to interpolate the points defined by the user.

Table 17 shows properties of the attributes of the element **pointSetMonotonicShape**.

Table 17—Properties of the attributes of the element pointSetMonotonicShape

Name	Use	Values	Default value
interpolationMethod	optional	Any string that conforms to the following patterns: – `linear` for implementing the linear interpolation as defined in [B11]; – `cubic` for implementing the monotone cubic interpolation as defined in [B9];	`linear`

The element **pointSetMonotonicShape** contains two or multiple elements **point**. The element **point** represents a point of the geometric area defining the monotonic fuzzy shape. It has two attributes (see Figure 8), as follows:

— The attribute **x** is used to define the coordinate on the x-axis.

— The attribute **y** is used to define the coordinate on the y-axis.

Table 9 shows properties of the attributes of the element **point**.

The element **customMonotonicShape** (see Figure 10) has an only attribute as follows:

— The attribute **name** is used to define a name for the custom shape.

Table 13 shows properties of the attributes of the element **customShape**.

The element **customMonotonicShape** contains zero or more elements **parameter**, which represents a parameter useful to define the custom shape for a fuzzy term. The element parameter (see Figure 10) has an only attribute as follows:

— The attribute **value** is used to define the value for a parameter characterizing the custom shape.

Table 14 shows properties of the attributes of the element **parameter**.

The element **aggregatedFuzzyVariable** (see Figure 12 and Figure 14) has three attributes as follows:

— The attribute **name** is used to define a unique name for the aggregated fuzzy variable.

— The attribute **type** is used to define the position of the aggregated fuzzy variable into rule (consequent part or antecedent part).

— The attribute **networkAddress** is used to define the location of the aggregated fuzzy variable in a computer network.

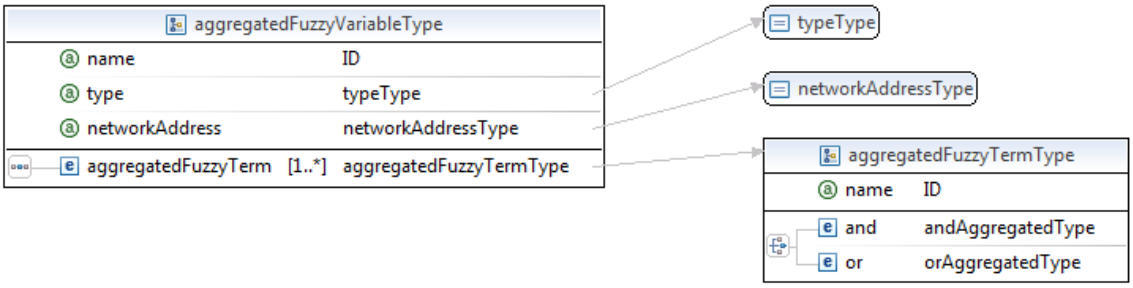


Figure 12—Type diagram for the elements aggregatedFuzzyVariable and aggregatedFuzzyTerm

Table 18 shows properties of the attributes of the element **aggregatedFuzzyVariable**.

Table 18—Properties of the attributes of the element aggregatedFuzzyVariable

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0" [B6].	—
type	required	The string `output`.	—
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	`127.0.0.1`

The element **aggregatedFuzzyVariable** contains one or multiple elements **aggregatedFuzzyTerm**, as shown in Figure 12. The element **aggregatedFuzzyTerm** (see Figure 12 and Figure 7) has an only attribute as follows:

— The attribute **name** is used to define a name for the term of the aggregated fuzzy variable.

Table 19 shows properties of the attributes of the element **aggregatedFuzzyTerm.**

Table 19—Properties of the attributes of the element aggregatedFuzzyTerm

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0" [B6].	—

The element **aggregatedFuzzyTerm** contains one element chosen from the following elements: **and**, **or** (see Figure 12). The element **and** has an only attribute (see Figure 13 and Figure 9):

— The attribute **t-norm** is used to define the method to be associated to the FL operator *and*.

Table 10 shows properties of the attributes of the element **and**. At the same way, the element **or** has an only attribute (see Figure 13):

— The attribute **t-conorm** is used to define the method to be associated to the FL operator *or*.

Table 11 shows properties of the attributes of the element **or**.

Both the element **and** and the element **or** contain two elements, **clause** or an element **and** followed by an element **clause** or an element **or** followed by an element **clause** (see Figure 13). The element **clause** represents the clause to be used in the fuzzy expression of the aggregated fuzzy term under definition. The element **clause** has an only attribute (see Figure 19), as follows:

— The attribute **modifier** is used to describe a modification to the linguistic term used in the clause.

Table 27 shows properties of the attributes of the element **clause**.

The elements **clause** contains an element **variable** and an element **term**, used respectively, to represent the variable and the term characterizing the fuzzy clause (see Figure 19).

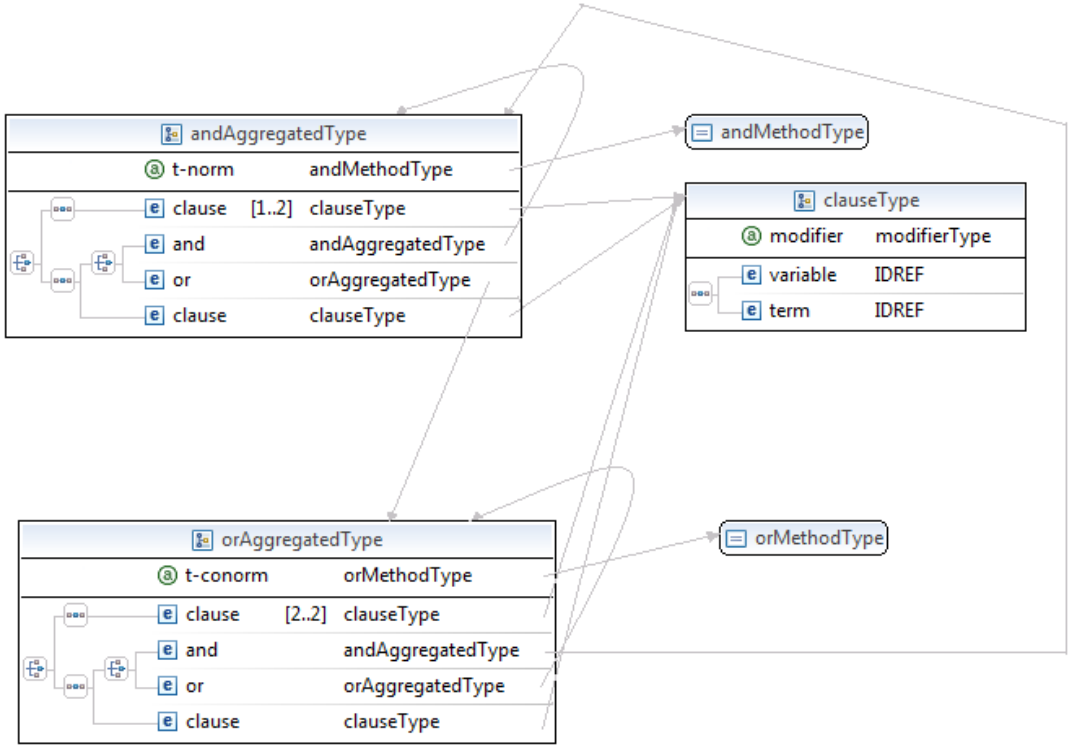


Figure 13—Type diagram for the elements and and or innested in the element aggreatedFyzzyTerm

The element **tskVariable** (see Figure 14) has six attributes as follows:

— The attribute **name** is used to define a unique name for a fuzzy variable.

— The attribute **scale** is used to define the scale used to measure a fuzzy variable.

— The attribute **type** is used to define the position of a fuzzy variable into rule (consequent part or antecedent part).

— The attribute **combination** is used to define the method of aggregation of the TSK rule outputs to obtain a final result.

— The attribute **defaultValue** is used to define a real value used only when no rule has fired for the variable at issue.

— The attribute **networkAddress** is used to define the location of the fuzzy variable in a computer network.

Table 20 shows properties of the attributes of the element **tskVariable**.

Table 20—Properties of the attributes of the element tskVariable

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0" [B6].	—
scale	optional	Any string.	Not provided
type	required	The string `output`.	—
combination	optional	Any string that conforms to the following patterns: – `WA` for the weighted average as defined from Equation (A.47); – `custom_\S*` for a custom aggregation of the TSK rule outputs. NOTE: Use of the last pattern results in a document *conformant* but not *strictly conformant*.	`WA`
defaultValue	optional	Any float.	`0`
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	`127.0.0.1`

The element **tskVariable** contains one or multiple elements **tskTerm**, as shown in Figure 14.

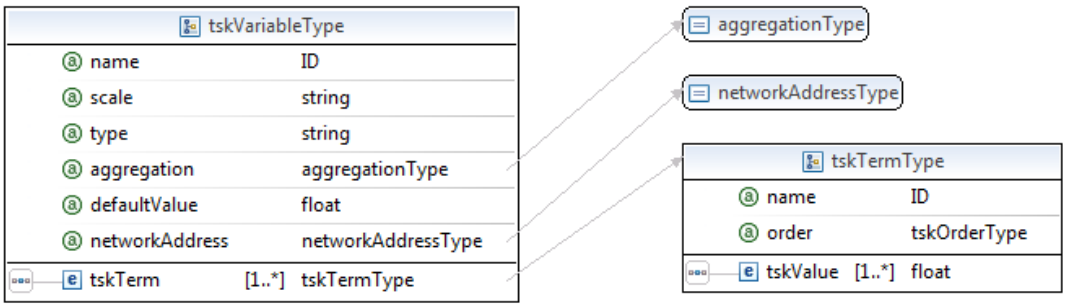


Figure 14—Type diagram for the elements tskVariable and tskTerm

The element **tskTerm** represents a constant value (order zero TSK system) or a linear function of system's inputs (order one TSK system). It has two attributes (see Figure 14) as follows:

— The attribute **name** is used to identify a unique name for the term associated with a fuzzy variable.

— The attribute **order** is used to describe the order of a TSK system.

Table 21 shows properties of the attributes of the element **tskTerm**.

Table 21—Properties of the attributes of the element tskTerm

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a NCName in the Recommendation "Namespaces in XML 1.0" [B6].	—
order	required	Any string that conforms to the following pattern: `(0|1).`	—

The element **tskTerm** contains one or multiple elements **tskValue** (see Figure 14) that represents a coefficient of the linear function characterizing the TSK term.

The element **anYaDataCloud** (see Figure 15) has two attributes as follows:

— The attribute **name** is used to univoquely identify the data cloud.

— The attribute **networkAddress** is used to define the location of the data cloud in a computer network.

Table 22 shows properties of the attributes of the element **anYaDataCloud**.

Table 22—Properties of the attributes of the element anYaDataCloud

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a NCName in the Recommendation "Namespaces in XML 1.0" [B6].	—
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	`127.0.0.1`

The element **anYaDataCloud** contains one or multiple elements **datum** (see Figure 15), each one of them represents a datum in the data cloud.

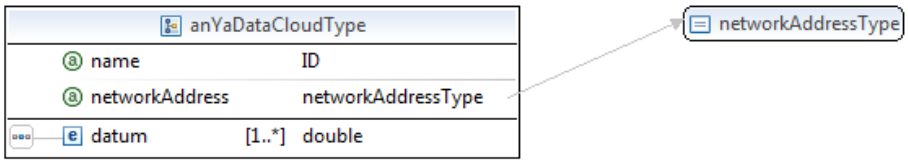


Figure 15—Type diagram for the element anYaDataCloud

The elements **mamdaniRuleBase**, **tsukamotoRuleBase**, **tskRuleBase**, and **anYaRuleBase** represent fuzzy rule sets. The element **fuzzySystem** may contain more than one element **mamdaniRuleBase**, and/or **tsukamotoRuleBase**, and/or **tskRuleBase**, and/or **anYaRuleBase** in order to describe different behaviors of the system. The elements **mamdaniRuleBase**, **tsukamotoRuleBase**, and **tskRuleBase** have five attributes (see Figure 5) as follows:

— The attribute **name** is used to uniquely identify the rule base.

— The attribute **activationMethod** is used to define the method used for the implication process.

— The attribute **andMethod** is used to define the *and* algorithm to be used by default for all rules.

— The attribute **orMethod** is used to define the *or* algorithm to be used by default for all rules.

— The attribute **networkAddress** is used to define the location of the rule base in a computer network.

Table 23 shows properties of the attributes of the elements **mamdaniRuleBase**, **tsukamotoRuleBase**, and **tskRuleBase**.

Table 23—Properties of the attributes of the elements mamdaniRuleBase, tsukamotoRuleBase, and tskRuleBase

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0" [B6].	—
activationMethod	optional	Any string that conforms to the following patterns: – `MIN` for implementing the implication with the minimum as defined from Equation (A.28); – `PROD` for implementing the implication with the product as defined from Equation (A.29); – `BDIF` for implementing the implication with bounded difference as defined from Equation (A.30); – `DRP` for implementing the implication with the drastic product as defined from Equation (A.31); – `EPROD` for implementing the implication with the Einstein product as defined from Equation (A.32); – `HPROD` for implementing the implication with the Hamacher product as defined from Equation (A.33); – `NILMIN` for implementing the implication with the Nilpotent minimum as defined from Equation (A.34); – `custom_\S*` for a custom implication method. See NOTE.	**`MIN`**
andMethod	optional	Any string that conforms to the following patterns: – `MIN` for implementing the operator *and* with the minimum as defined from Equation (A.14); – `PROD` for implementing the operator *and* with the product as defined from Equation (A.15); – `BDIF` for implementing the operator *and* with bounded difference as defined from Equation (A.16); – `DRP` for implementing the operator *and* with the drastic product as defined from Equation (A.17); – `EPROD` for implementing the operator *and* with the Einstein product as defined from Equation (A.18); – `HPROD` for implementing the operator *and* with the Hamacher product as defined from Equation (A.19); – `NILMIN` for implementing the operator *and* with the Nilpotent minimum as defined from Equation (A.20); – `custom_\S*` for a custom method for operator *and*. See NOTE	**`MIN`**

Table continues

Table 23—Properties of the attributes of the elements mamdaniRuleBase, tsukamotoRuleBase, and tskRuleBase *(continued)*

Name	Use	Values	Default value
orMethod	optional	Any string that conforms to the following patterns: – `MAX` for implementing the connector *or* with the maximum as defined from Equation (A.21); – `PROBOR` for implementing the connector *or* with the probabilistic sum as defined from Equation (A.22); – `BSUM` for implementing the operator *or* with the bounded sum as defined from Equation (A.23); – `DRS` for implementing the operator *or* with the drastic sum as defined from Equation (A.24); – `ESUM` for implementing the operator or with the Einstein sum as defined from Equation (A.25); – `HSUM` for implementing the operator *or* with the Hamacher sum as defined from Equation (A.26); – `NILMAX` for implementing the operator *or* with the Nilpotent maximum as defined from Equation (A.27); – `custom_\S*` for a custom method for implementing the connector *or*. See NOTE.	`MAX`
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	`127.0.0.1`
NOTE—Use of the last pattern results in a document *conformant* but not *strictly conformant*.			

The element **anYaRuleBase** has three attributes (see Figure 5) as follows:

— The attribute **name** is used to uniquely identify the rule base.

— The attribute **activationMethod** is used to define the method used for the implication process.

— The attribute **networkAddress** is used to define the location of the rule base in a computer network.

Table 24 shows properties of the attributes of the element **anYaRuleBase**.

The elements **mamdaniRuleBase** and **tsukamotoRuleBase** contain one or more elements **rule** (see Figure 5). The element **tskRuleBase** contains one or more element **tskRule** (see Figure 5). The element **anYaRuleBase** contains one or more element **anYaRule** (see Figure 5).

The element **rule** representing an IF-THEN-ELSE rule whose consequent part could support the exploitation of hedges, whereas the element **tskRule** representing an IF-THEN-ELSE rule whose consequent part does not allow defining hedges. Both elements **rule** and **tskRule** have six attributes (see Figure 16), as follows:

— The attribute **name** is used to uniquely identify the rule.

— The attribute **connector** is used to define the logical operator aimed at connecting the different clauses in antecedent part (*and*/*or*).

— The attribute **andMethod** is used to define the *and* algorithm to be used if the chosen connector is *and*. The value of this attribute overrides the value of the attributes **andMethod** of the elements **mamdaniRuleBase**, **tsukamotoRuleBase**, and **tskRuleBase**.

— The attribute **orMethod** is used to define the *or* algorithm to be used if the chosen connector is *or*. The value of this attribute overrides the value of the attributes **orMethod** of the elements **mamdaniRuleBase**, **tsukamotoRuleBase**, and **tskRuleBase**.

— The attribute **weight** is used to define the importance of the rule to be used by the inference engine.

— The attribute **networkAddress** is used to define the location of the rule in a computer network.

Table 24—Properties of the attributes of the element anYaRuleBase

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0" [B6].	—
activationMethod	optional	Any string that conforms to the following patterns: – `MIN` for implementing the implication with the minimum as defined from Equation (A.28); – `PROD` for implementing the implication with the product as defined from Equation (A.29); – `BDIF` for implementing the implication with bounded difference as defined from Equation (A.30); – `DRP` for implementing the implication with the drastic product as defined from Equation (A.31); – `EPROD` for implementing the implication with the Einstein product as defined from Equation (A.32); – `HPROD` for implementing the implication with the Hamacher product as defined from Equation (A.33); – `NILMIN` for implementing the implication with the Nilpotent minimum as defined from Equation (A.34); – `custom_\S*` for a custom implication method. See NOTE.	`MIN`
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `[0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	`127.0.0.1`
NOTE—Use of the last pattern results in a document *conformant* but not *strictly conformant*.			

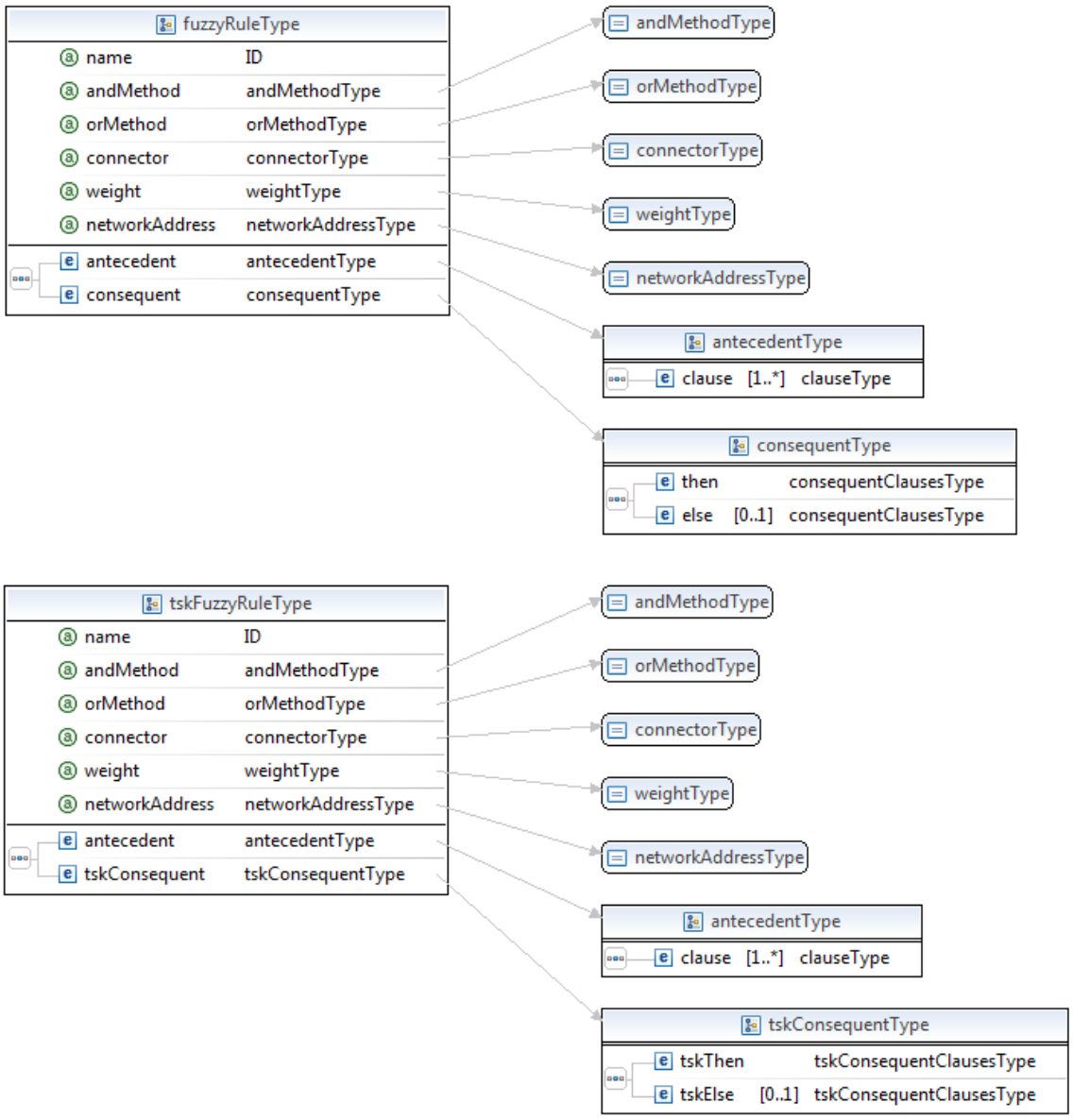


Figure 16—Type diagram for the elements rule and tskRule

Table 25 shows properties of the attributes of the elements **rule** and **tskRule**.

The element **anYaRule** represents an IF-THEN-ELSE rule of a data cloud-based fuzzy system. It has three attributes (see Figure 17), as follows:

— The attribute **name** is used to uniquely identify the rule.

— The attribute **weight** is used to define the importance of the rule to be used by the inference engine.

— The attribute **networkAddress** is used to define the location of the rule in a computer network.

Table 25—Properties of the attributes of the elements rule and tskRule

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0"[B6].	—
connector	optional	Any string that conforms to the following pattern: `(and|or|AND|OR)`.	`and`
andMethod	optional	Any string that conforms to the following patterns: – `MIN` for implementing the operator *and* with the minimum as defined from Equation (A.14); – `PROD` for implementing the operator *and* with the product as defined from Equation (A.15); – `BDIF` for implementing the operator *and* with bounded difference as defined from Equation (A.16); – `DRP` for implementing the operator *and* with the drastic product as defined from Equation (A.17); – `EPROD` for implementing the operator *and* with the Einstein product as defined from Equation (A.18); – `HPROD` for implementing the operator *and* with the Hamacher product as defined from Equation (A.19); – `NILMIN` for implementing the operator *and* with the Nilpotent minimum as defined from Equation (A.20); – `custom_\S*` for a custom method for operator *and*. NOTE: Use of the last pattern results in a document *conformant* but not *strictly conformant*.	`MIN`
orMethod	optional	Any string that conforms to the following patterns: – `MAX` for implementing the connector *or* with the maximum as defined from Equation (A.21); – `PROBOR` for implementing the connector *or* with the probabilistic sum as defined from Equation (A.22); – `BSUM` for implementing the operator *or* with the bounded sum as defined from Equation (A.23); – `DRS` for implementing the operator *or* with the drastic sum as defined from Equation (A.24); – `ESUM` for implementing the operator or with the Einstein sum as defined from Equation (A.25); – `HSUM` for implementing the operator *or* with the Hamacher sum as defined from Equation (A.26); – `NILMAX` for implementing the operator *or* with the Nilpotent maximum as defined from Equation (A.27); – `custom_\S*` for a custom method for implementing the connector *or*. NOTE: Use of the last pattern results in a document *conformant* but not *strictly conformant*.	`MAX`
weight	optional	Any float in the range [0,1].	`1.0`
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	`127.0.0.1`

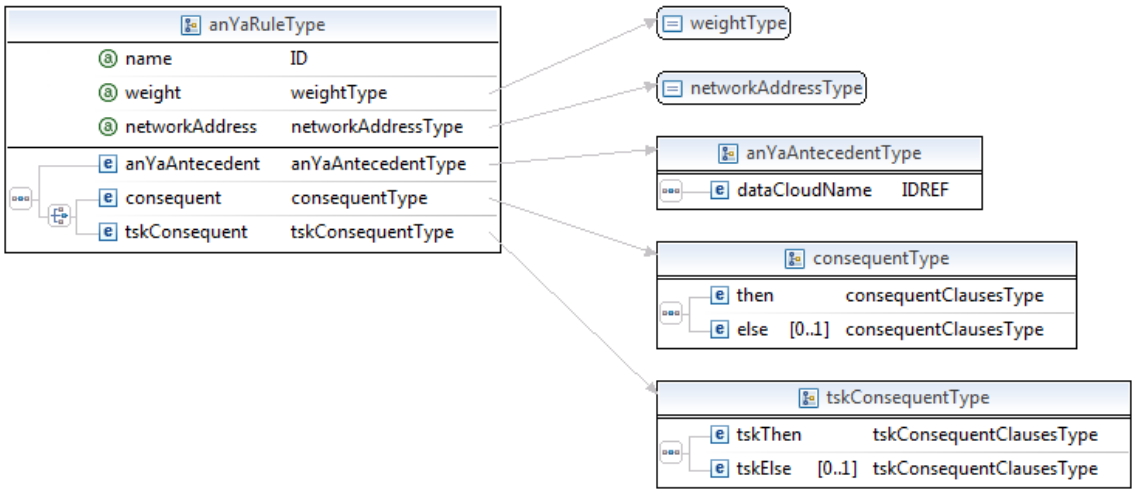


Figure 17—Type diagram for the elements anYaRule and anYaAntecedent

Table 26 shows properties of the attributes of the element **anYaRule**.

Table 26—Properties of the attributes of the element anYaRule

Name	Use	Values	Default value
name	required	Any string that conforms to the definition of a `NCName` in the Recommendation "Namespaces in XML 1.0" [B6].	—
weight	optional	Any float in the range [0,1].	`1.0`
networkAddress	optional	Any string that conforms to the following patterns: – `(([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])\.){3}([0-1]?[0-9]?[0-9]|2?[0-5]?[0-5])` for an address IPv4 as defined in [B18][B18] at pages 7 – 8; – `(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))` for an address IPv6 as defined in [B18] at page 8; – `([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}` for a mac address; – `[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)*` for a host name.	`127.0.0.1`

The element **rule** contains one element **antecedent** and one element **consequent** as shown in Figure 16. The element **tskRule** contains one element **antecedent** and one element **tskConsequent** as shown in Figure 16. The element **anYaRule** contains one element **anYaAntecedent** followed by one element **consequent** or one element **tskConsequent** as shown in Figure 17.

The element **antecedent** represents the antecedent part of a rule based on the aggregation of conventional fuzzy clauses, whereas the **anYaAntecedent** represents the antecedent part of a rule based on data clouds.

The element **consequent** represents the THEN[-ELSE] part of a rule based on fuzzy clauses, whereas the element **tskConsequent** represents the THEN[-ELSE] part of a rule in an TSK system (constant or linear function).

The element **antecedent** contains one or multiple elements **clause**, each one of them representing a Mamdani clause (see Figure 16). The element **consequent** contains one element **then,** representing the THEN-part of a rule, and an optional element **else**, representing the ELSE-part of a rule (see Figure 16). The element **tskConsequent** contains one element **tskThen,** representing the THEN-part of a TSK rule, and an optional element **tskElse**, representing the ELSE-part of a TSK rule (see Figure 16). The elements **then** and **else** contain one or multiple elements **clause,** which represents a Mamdani clause (see Figure 18). The elements **tskThen** and **tskElse** contain one or multiple elements **tskClause**, which represents a TSK clause (see Figure 18). The element **anYaAntecedent** contains one element **dataCloudName,** which represents the name of the data cloud to be used in the rule (see Figure 17).

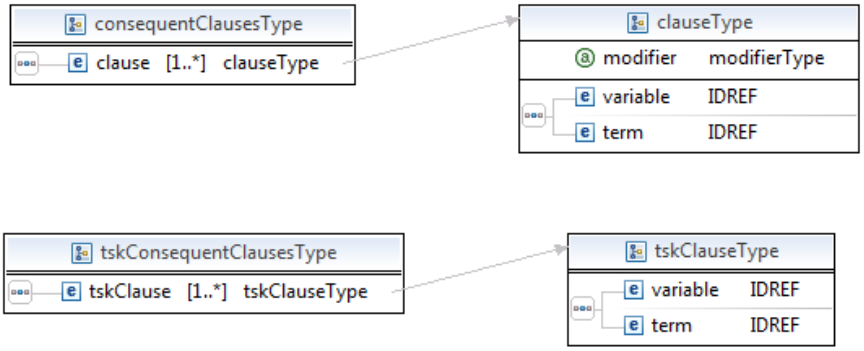


Figure 18—Type diagram for the elements then, else, tskThen and tskElse

The element **clause** has an only attribute (see Figure 19), as follows:

— The attribute **modifier** is used to describe a modification to the linguistic term used in the clause.

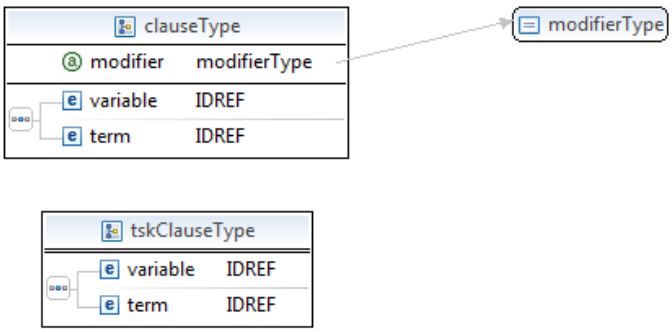


Figure 19—Type diagram for the elements clause and tskClause

Table 27 shows properties of the attributes of the element **clause**.

Table 27—Properties of the attributes of the element clause

Name	Use	Values	Default value
modifier	optional	Any string that conforms to the following patterns: – `above` for the hedge *above* as defined from Equation (A.1); – `any` for the hedge *any* as defined from Equation (A.2); – `below` for the hedge *below* as defined from Equation (A.3); – `extremely` for the hedge *extremely* as defined from Equation (A.4); – `intensify` for the hedge *intesify* as defined from Equation (A.5); – `more_or_less` for the hedge *more or less* as defined from Equation (A.6); – `norm` for the hedge norm as defined from Equation (A.7); – `not` for the hedge *not* as defined from Equation (A.8); – `plus` for the hedge *plus* as defined from Equation (A.9); – `seldom` for the hedge *seldom* as defined from Equation (A.10); – `slightly` for the hedge *slightly* as defined from Equation (A.11); – `somewhat` for the hedge *somewhat* as defined from Equation (A.12); – `very` for the hedge *very* as defined from Equation (A.13); – `custom_\S*` for a custom hedge.	Not provided
NOTE—Use of the last pattern results in a document *conformant* but not *strictly conformant*.			

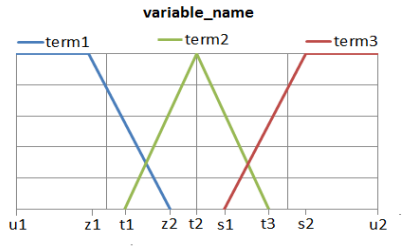
The elements **clause** and **tskClause** contain an element **variable** and an element **term**, used respectively, to represent the variable and the term characterizing the fuzzy clause (see Figure 19).

5.2.2 Examples

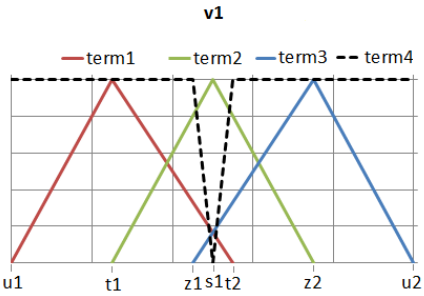
In order to better understand the FML syntax, we present some examples that show how this standard models the main components of a FLS and a specific kind of fuzzy systems. For complete examples, see Annex C.

5.2.2.1 Input variable examples

Usual Description	FML Description


An input variable *variable_name* with three terms *term1*, *term2* and *term3*.
	`<fuzzyVariable name="variable_name" domainleft="u1" domainright="u2" scale="" type="input">`
	`  <fuzzyTerm name="term1" complement="false">`
	`  <zShape param1="z1" param2="z2"/>`
	`  </fuzzyTerm>`
	`  <fuzzyTerm name="term2" complement="false">`
	`  <triangularShape param1="t1" param2="t2" param3="t3"/>`
	`  </fuzzyTerm>`
	`  <fuzzyTerm name="term3" complement="false">`
	`  <sShape param1="s1" param2="s2"/>`
	`  </fuzzyTerm>`
	`</fuzzyVariable>`

Usual Description	FML Description

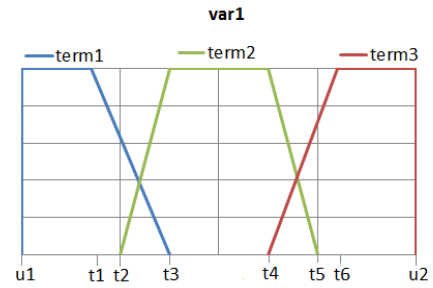
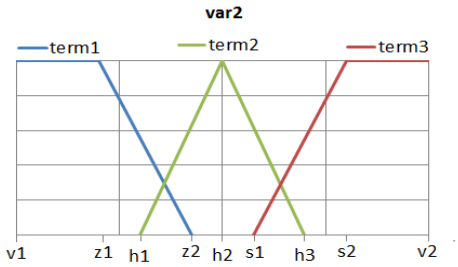
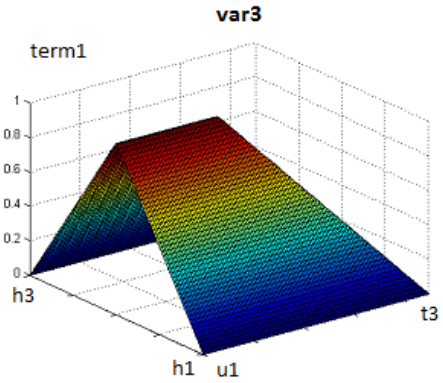
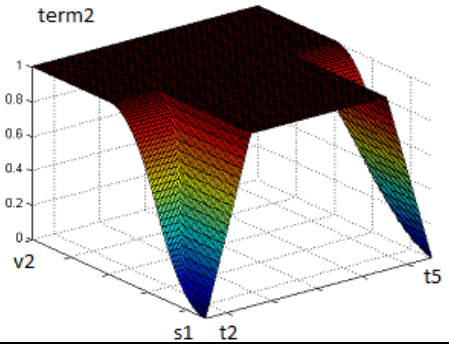

An input variable *v1* with four terms *term1*, *term2*, *term3* and *term4* where *term4* is computed as follows:
$term4 = \neg(term1 \wedge term2 \wedge term3)$
	`<fuzzyVariable name="variable_name"`
	`domainleft="u1" domainright="u2" scale=""`
	`type="input">`
	`  <fuzzyTerm name="term1" complement="false">`
	`  <triangularShape param1="u1" param2="t1"`
	`  param3="t2"/>`
	`  </fuzzyTerm>`
	`  <fuzzyTerm name="term2" complement="false">`
	`  <triangularShape param1="t1" param2="s1"`
	`  param3="z2"/>`
	`  </fuzzyTerm>`
	`  <fuzzyTerm name="term3" complement="false">`
	`  <triangularShape param1="z1" param2="z2"`
	`  param3="u2"/>`
	`  </fuzzyTerm>`
	`  <fuzzyTerm name="term4" complement="true">`
	`  <circularDefinition>`
	`  <and t-norm="MIN">`
	`  <and t-norm="MIN">`
	`  <termName>term1</termName>`
	`  <termName>term2</termName>`
	`  </and>`
	`  <termName>term3</termName>`
	`  </and>`
	`  </circularDefinition>`
	`  </fuzzyTerm>`
	`</fuzzyVariable>`

Usual Description	FML Description

cloud1=[*p1*, *p2*, *p3*]	`<anYaDataCloud name="cloud1">`
	`  <datum>p1</datum>`
	`  <datum>p2</datum>`
	`  <datum>p3</datum>`
	`</anYaDataCloud>`

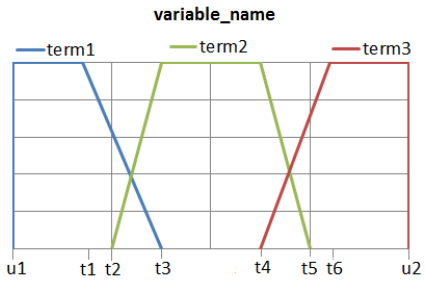
A data cloud, named *cloud1*, composed of three data *p1*, *p2*, *p3*.	

Usual Description	FML Description

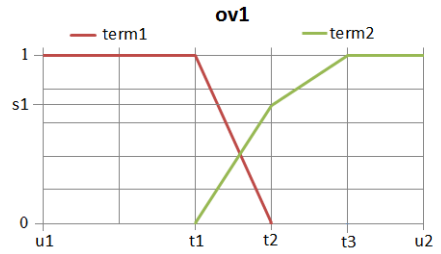

 
 
 
An aggregated fuzzy variable named *var3* with two terms named *term1* and *term2* where *term1* is computed as follows:
term1 = *var1* is *term1* and *var2* is *term2*
and *term2* is computed as follows:
term2 = *var1* is *term2* or *var2* is *term3*
	`<fuzzyVariable name="var1" domainleft="u1" domainright="u2" scale="" defaultValue="d" type="input">`
	`<fuzzyTerm name="term1" complement="false">`
	`<trapezoidShape param1="u1" param2="u1" param3="t1" param4="t3"/>`
	`</fuzzyTerm>`
	`<fuzzyTerm name="term2" complement="false">`
	`<trapezoidShape param1="t2" param2="t3" param3="t4" param4="t5"/>`
	`</fuzzyTerm>`
	`<fuzzyTerm name="term3" complement="false">`
	`<trapezoidShape param1="t4" param2="t6" param3="u2" param4="u2"/>`
	`</fuzzyTerm>`
	`</fuzzyVariable>`
	`<fuzzyVariable name="var2" domainleft="v1" domainright="v2" scale="" type="input">`
	`<fuzzyTerm name="term1" complement="false">`
	`<zShape param1="z1" param2="z2"/>`
	`</fuzzyTerm>`
	`<fuzzyTerm name="term2" complement="false">`
	`<triangularShape param1="h1" param2="h2" param3="h3"/>`
	`</fuzzyTerm>`
	`<fuzzyTerm name="term3" complement="false">`
	`<sShape param1="s1" param2="s2"/>`
	`</fuzzyTerm>`
	`</fuzzyVariable>`
	`<aggregatedFuzzyVariable name="var3" type="input">`
	`<aggregatedFuzzyTerm name="term1">`
	`<and t-norm="MIN">`
	`<clause>`
	`<variable>v1</variable>`
	`<term>term1</term>`
	`</clause>`
	`<clause>`
	`<variable>v2</variable>`
	`<term>term2</term>`
	`<clause>`
	`</and>`
	`</aggregatedFuzzyTerm>`
	`<aggregatedFuzzyTerm name="term2">`
	`<or t-conorm="MAX">`
	`<clause>`
	`<variable>v1</variable>`
	`<term>term2</term>`
	`</clause>`
	`<clause>`
	`<variable>v2</variable>`
	`<term>term3</term>`
	`<clause>`
	`</or>`
	`</aggregatedFuzzyTerm>`
	`</aggregatedFuzzyVariable>`

5.2.2.2 Output variable examples

Usual Description	FML Description


An output variable *variable_name* with three terms *term1*, *term2* and *term3*. The variable is characterized by a default value *d*, an accumulation method *acc_method* and a defuzzifier method *def_method*.
	`<fuzzyVariable name="variable_name"`
	`domainleft="u1" domainright="u2" scale=""`
	`defaultValue="d" accumulation="acc_method"`
	`defuzzifier="def_method" type="output">`
	`<fuzzyTerm name="term1" complement="false">`
	`<trapezoidShape param1="u1" param2="u1"`
	`param3="t1" param4="t3"/>`
	`</fuzzyTerm>`
	`<fuzzyTerm name="term2" complement="false">`
	`<trapezoidShape param1="t2" param2="t3"`
	`param3="t4" param4="t5"/>`
	`</fuzzyTerm>`
	`<fuzzyTerm name="term3" complement="false">`
	`<trapezoidShape param1="t4" param2="t6"`
	`param3="u2" param4="u2"/>`
	`</fuzzyTerm>`
	`</fuzzyVariable>`

Usual Description	FML Description


A tsukamoto variable *ov1* with two terms *term1* and *term2*. The variable is characterized by a default value *d* and by the combination method *Weighted Average* (WA). The points characterizing the term *term2* are interpolated through a linear method.
	`<tsukamotoVariable name="ov1"`
	`domainleft="u1" domainright="u2" scale=""`
	`defaultValue="d" combination="WA"`
	`type="output">`
	`<tsukamotoTerm name="term1"`
	`complement="false">`
	`<zShape param1="t1" param2="t2"/>`
	`</tsukamotoTerm>`
	`<tsukamotoTerm name="term2"`
	`complement="false">`
	`<pointSetMonotonicShape`
	`interpolationMethod="linear" >`
	`<point x="t1" y="0"/>`
	`<point x="t2" y="s1"/>`
	`<point x="t3" y="1"/>`
	`<point x="u2" y="1"/>`
	`</pointSetMonotonicShape>`
	`</tsukamotoTerm>`
	`</tsukamotoVariable>`

5.2.2.3 Rule examples

Usual Description	FML Description

reg1: **IF** *v1* is *term1* **or** *v2* is *term2* **THEN** *v3 is term3* (1.0)
A rule of a Mamdani system with two input variables *v1* and *v2* and an output variables *v3*. The rule, named *reg1*, is characterized by the connector **or**, the operator *op* used for implementing the connector **or** and a weight that is equal to 1.0.
	`<rule name="reg1" connector="or"`
	`orMethod="op" weight="1.0">`
	`<antecedent>`
	`<clause>`
	`<variable>v1</variable>`
	`<term>term1</term>`
	`</clause>`
	`<clause>`
	`<variable>v2</variable>`
	`<term>term2</term>`
	`</clause>`
	`</antecedent>`
	`<consequent>`
	`<then>`
	`<clause>`
	`<variable>v3</variable>`
	`<term>term3</term>`
	`</clause>`
	`</then>`
	`</consequent>`
	`</rule>`

Usual Description	FML Description

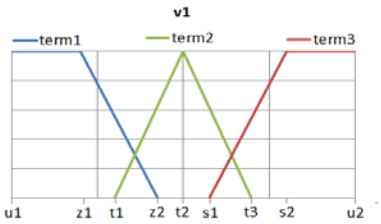
reg1: **IF** *v1* is *term1* **and** *v2* is *term2*
THEN *v3 is term3*
ELSE *v4 is term4* (1.0)
 	
A rule of a Mamdani system with two input variables *v1* and *v2* and two output variables *v3* and *v4*. The rule, named *reg1*, is characterized by the connector **and**, the operator *op* used for implementing the connector **and** and a weight that is equal to 1.0.
	`<rule name="reg1" connector="and" andMethod="op" weight="1.0">`
	`<antecedent>`
	`<clause>`
	`<variable>v1</variable>`
	`<term>term1</term>`
	`</clause>`
	`<clause>`
	`<variable>v2</variable>`
	`<term>term2</term>`
	`</clause>`
	`</antecedent>`
	`<consequent>`
	`<then>`
	`<clause>`
	`<variable>v3</variable>`
	`<term>term3</term>`
	`</clause>`
	`</then>`
	`<else>`
	`<clause>`
	`<variable>v4</variable>`
	`<term>term4</term>`
	`</clause>`
	`</else>`
	`</consequent>`
	`</rule>`

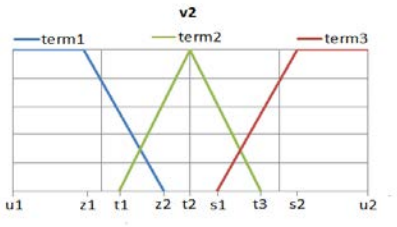
Usual Description	FML Description

reg1: **IF x** ~ *cloud1*
THEN *v3 is term3* (1.0)
 	
A rule of an AnYa system related to the data cloud *cloud1* and characterized by a Mamdani consequent part involving the variable *v3*. **x** represents the input vector. The rule, named *reg1*, has a weight that is equal to 1.0.
	`<anYaRule name="reg1" weight="1.0">`
	`<anYaAntecedent>`
	`<dataCloudName>cloud1</dataCloudName>`
	`</anYaAntecedent>`
	`<consequent>`
	`<then>`
	`<clause>`
	`<variable>v3</variable>`
	`<term>term3</term>`
	`</clause>`
	`</then>`
	`</consequent>`
	`</anYaRule>`

5.2.2.4 TSK system example

Usual Description





reg1: **IF** *v1* is *term1* **and** *v2* is *term2* **THEN** $y=c+av1+bv2$ (1.0)

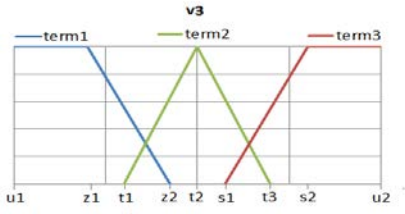
A TSK system, named *tsk_name*, with two input variables *v1* and *v2* and one output variable *y*. The variable *y* is characterized by the combination method WA. The rule base, named *rulebase_name*, is characterized by the activation method *act_method* and it is composed of the rule, named *reg1*, characterized by the operator *op* for implementing the connector **and**. The coefficients ***c***, ***a***, ***b*** are organized in a term, named *tsk_term*. Due to the linear function characterizing the rule *reg1*, the system is of order 1.

FML Description

```
<fuzzySystem name="tsk_name">
  <knowledgeBase>
    <fuzzyVariable name="v1" domainleft="u1"
                domainright="u2" scale=""
                type="input">
      <fuzzyTerm name="term1" complement="false">
        <zShape param1="z1" param2="z2"/>
      </fuzzyTerm>
      <fuzzyTerm name="term2" complement="false">
        <triangularShape param1="t1" param2="t2"
                                    param3="t3"/>
      </fuzzyTerm>
      <fuzzyTerm name="term3" complement="false">
        <sShape param1="s1" param2="s2"/>
      </fuzzyTerm>
    </fuzzyVariable>
    <fuzzyVariable name="v2" domainleft="u1"
                domainright="u2" scale=""
                type="input">
      <fuzzyTerm name="term1" complement="false">
        <zShape param1="z1" param2="z2"/>
      </fuzzyTerm>
      <fuzzyTerm name="term2" complement="false">
        <triangularShape param1="t1" param2="t2"
                                    param3="t3"/>
      </fuzzyTerm>
      <fuzzyTerm name="term3" complement="false">
        <sShape param1="s1" param2="s2"/>
      </fuzzyTerm>
    </fuzzyVariable>
    <tskVariable name="y" scale="null"
            combination="WA" type="output">
      <tskTerm name="tsk_term" order="1">
        <tskValue>c</tskValue>
        <tskValue>a</tskValue>
        <tskValue>b</tskValue>
      </tskTerm>
    </tskVariable>
  </knowledgeBase>
  <tskRuleBase name="rulebase_name"
            activationMethod="act_method">
    <tskRule name="reg1" connector="and"
          andMethod="op" weight="1.0">
      <antecedent>
        <clause>
          <variable>v1</variable>
          <term>term1</term>
        </clause>
        <clause>
          <variable>v2</variable>
          <term>term2</term>
        </clause>
      </antecedent>
      <tskConsequent>
        <tskThen>
          <tskClause>
            <variable>y</variable>
            <term>tsk_term</term>
          </tskClause>
        </tskThen>
      </tskConsequent>
    </tskRule>
  </tskRuleBase>
</fuzzySystem>
```

5.2.2.5 Standard Additive Model (SAM) system example

Usual Description



reg1: **IF** *v1* is *term1* **and** *v2* is *term2* **THEN** *v3* is *term3* (1.0)

A SAM system, named *sam_name*, with two input variables *v1* and v*2* and an output variable *v3*. The variable *v3* is characterized by the accumulation method *SUM* and by the defuzzifier center of gravity (COG). The rule base, named *rulebase_name*, is characterized by the activation method *PROD* and it is composed of the rule, named *reg1*, characterized by the operator *product* (PROD) for implementing the connector *and*.

FML Description

```
<fuzzySystem name="sam_name">
  <knowledgeBase>
   …
    <fuzzyVariable name="v3" domainleft="u1"
                domainright="u2" scale=""
                type="output" defuzzifier="COG"
                accumulation="SUM">
      <fuzzyTerm name="term1" complement="false">
        <zShape param1="z1" param2="z2"/>
      </fuzzyTerm>
      <fuzzyTerm name="term2" complement="false">
        <triangularShape param1="t1" param2="t2"
                                     param3="t3"/>
      </fuzzyTerm>
      <fuzzyTerm name="term3" complement="false">
        <sShape param1="s1" param2="s2"/>
      </fuzzyTerm>
    </fuzzyVariable>
  </knowledgeBase>
  <mamdaniRuleBase name="rulebase_name"
            activationMethod="PROD">
    <rule name="reg1" connector="and"
          andMethod="PROD" weight="1.0">
      <antecedent>
        <clause>
          <variable>v1</variable>
          <term>term1</term>
        </clause>
        <clause>
          <variable>v2</variable>
          <term>term2</term>
        </clause>
      </antecedent>
      <consequent>
        <then>
          <clause>
            <variable>v3</variable>
            <term>term3</term>
          </clause>
        </then>
      </consequent>
    </rule>
  </mamdaniRuleBase>
</fuzzySystem>
```

6. FML Synthesis: XSLT Documents and DOM API

The XML-based nature of the language defined in this standard allows providing the capability to generate executable programs in a direct way. In particular, the so-called eXtensible Stylesheet Language Translators (XSLTs) can be used to design so-called FML drivers aimed at converting an FML code into a general-purpose computer language program to be run on several hardware platforms. Precisely, XSLTs can be used for converting FML programs in legacy languages related to a particular hardware or in general purpose languages such as Java.

Moreover, FML document can be accessed by different computer programming languages through the Document Object Model (DOM), a cross-platform and language-independent convention for representing and interacting with objects in XML documents. In this way, C/C++, Python, or Java programs can “read” fuzzy information from an FML document and they can istantiate the right objects (or procedure) to run the related fuzzy system.

7. Compatibility with IEC 61131-7

IEC 61131 is a family of standards published by the International Electrotechnical Commission (IEC) as a result of an emerging need of the international industry community to uniform programmable controllers in terms of hardware design, installation, testing, documentation, programming, and communication. Part 7 of IEC 61131 [B14] gives specifications for the Fuzzy Control Language (FCL), which is the current most popular language for Fuzzy Control Programming. FCL is one of the most important attempts of a high-level and specific purpose language aimed at modeling fuzzy systems in a hardware independent way [B7]. However, even though FCL enabled systems' designers to program their fuzzy artifacts in a direct way, without having some knowledge about conventional general purpose programming languages, FCL lacks in the following various aspects:

— FCL does not support hedges, does not support ELSE-part of a rule, does not allow different operator definitions for the rules, and does not give support for default membership functions.

— FCL does not support a direct validation. Indeed, thanks to the benefits provided by the associated XML Schema, FML-based system can be validated as compliant with the related standard without effort.

— FCL does not enable flexible extensions. Thanks to some powerful features provided by XML, FML users can use customized fuzzy operators and rule base without changing the language grammar.

— FCL does not allow data binding and does not enable a direct generation of executable programs due to its pure textual nature. XML provides some technologies, such as DOM and XSLT, which enable FML users to convert their FML systems in runnable versions on different hardware platform in a simple and not expensive in terms of time way.

— FCL is based on a centralized approach and it does not provide, natively, any mechanism for modeling distributed fuzzy systems and, as a consequence, FCL programs cannot be directly computed in distributed computing scenarios. The hierarchical nature of FML, based on the concept of labeled tree, allows FML users to split their fuzzy models in different sub-hierarchies and deploy them in a computer network system.

The FML language specified in this standard covers all FCL components (as shown in Table 28) and, moreover, it overcomes the aforementioned drawbacks as a result of its XML based nature.

Table 28—Mapping between FCL and FML

FCL components		FML tags	
FCL Name	**Structure**	**FML Name**	**Structure**
Fuzzification	FUZZIFY *variable_name* TERM *term_name* := points; … END_FUZZIFY	Input Variable definition	<fuzzyVariable name="***variable_name***" domainleft="" domainright="" scale="" type="**input**"> <fuzzyTerm name="***term_name***" complement="">… </fuzzyTerm> … </fuzzyVariable>
Defuzzification	DEFUZZIFY *variable_name* TERM *term_name* := points; ACCU : *accumulation_method;* METHOD : *defuzzifier* DEFAULT := *value* [range ;] END_DEFUZZIFY	Output Variable definition	<fuzzyVariable name="***variable_name***" domainleft="" domainright="" scale="" defaultValue="***value***" accumulation="***accumulation_method***" defuzzifier="***defuzzifier***" type="**output**"> <fuzzyTerm name="***term_name***" complement="">… </fuzzyTerm> … </fuzzyVariable>
Rule block	RULEBLOCK *ruleblock*_name operator_definition ::= *operator* : *algorithm* [ACT : *activation_method;*] RULE *number*: IF condition THEN conclusion [WITH *weight*]; … END_RULEBLOCK	Rule base	<ruleBase name="***ruleblock_name***" andMethod="" orMethod="" activationMethod=" ***activation_method***"> <rule name="***number***" connector="***operator***" andMethod="**andAlgorithm**" orMethod="**orAlgorithm**" weight="***weight***"> <antecedent> <clause modifier=""> <variable>…</variable> <term>…</term> </clause> </antecedent> <consequent> <then> <clause modifier=""> <variable>…</variable> <term>…</term> </clause> </then> </consequent> </rule> … </ruleBase>

Table continues

Table 28—Mapping between FCL and FML *(continued)*

Function block	FUNCTION_BLOCK *function_block_name* VAR_INPUT *variable1_name*: *data_type*; … END_VAR VAR_OUTPUT *variable2_name*: *data_type*; … END_VAR … VAR *variable3_name*: *data_type*; END_VAR FUZZIFY *variable1_name* … END_FUZZIFY DEFUZZIFY *variable2_name* … END_DEFUZZIFY RULEBLOCK *ruleblock_name* … END_RULEBLOCK END_FUNCTION_BLOCK	Fuzzy System Definition	<fuzzySystem name=" ***function_block_name***" networkAddress="">… <knowledgeBase networkAddress="">… <fuzzyVariable name="***variable1_name***" domainleft="" domainright="" scale="" type="**input**">… </fuzzyVariable> <fuzzyVariable name="***variable2_name***" domainleft="" domainright="" scale="" defaultValue="***value***" accumulation= "***accumulation_method***" defuzzifier="***defuzzifier***" type="**output**">… </fuzzyVariable> </knowledgebase> <ruleBase name="***ruleblock_name***" andMethod="" orMethod="" activationMethod=" ***activation_method***">… </ruleBase> </fuzzySystem>

8. Conformance

Strictly conforming FML instance documents and *conforming* FML instance documents are discussed in this section.

A *strictly conforming* FML instance document:

- Shall be strictly complied with the FML Schema defined in Annex B
- Shall conform to the requirements of Clause 5
- Shall not include XML elements or attributes that are not defined in Clause 5
- Shall not include the element **customShape** and all its nested elements
- Shall not include attribute values that are not defined in Clause 5
- Shall not include attribute values that are defined by using the type *customLabelType* (i.e., defined by using the pattern [`custom_\S*`])

A *conforming* FML instance document:

- Shall be complied with the FML schema defined in Annex B
- Shall conform to the requirements of Clause 5
- May include XML elements that are not defined in Clause 5 but by using the wildcard-based extension mechanism described in Clause 9
- May include the element **customShape** and all its nested elements
- May include attribute values that are not defined in Clause 5 but by using the type *customLabelType* (i.e., defined by using the pattern [`custom_\S*`])

It is important to note that the fully interoperability is achievable only by respecting a strict conformance because conforming FML instance documents require to be managed in a customized way to be converted in the target language.

FML instance documents that are not *strictly conforming* or *conforming* are not conforming to this standard.

9. Extensibility

A provision in the XML schema of an extension mechanism is necessary to help ensure the viability of the specification and allow producers and consumers of FML instance documents to model higher order FLSs, and kinds of FLSs, designed in the future, or new fuzzy operators and hedges.

The use of the extensions shall be done in a way that helps ensure that a conformant consumer can utilize the extended file without error, discard, or otherwise sidestep the extended data and use the non-extended portions of the data as it is intended—without error or loss of functionality.

An extended instance document shall be accompanied by the extension FML schema and documentation sufficient to explain the need for the extension as well as the underlying semantics and relationship(s) to the base schema. This document shall be forwarded to the IEEE 1855 committee for potential inclusion in a later release.

Extensions to the FML schema shall only be used to express information that is not currently described in this standard.

Extensions shall not repackage existing information entities that are already supported by this standard.

Extensions shall always be associated with a user-defined namespace and should be identified with a namespace prefix.

FML schema supports two forms of extension:

— Inserting of additional elements in the element **fuzzySystem** and the element **knowedgeBase** through the wildcard-based extension mechanism. In particular, only elements from a namespace different from the document namespace are allowed for this extension. To meet this requirement, the following instruction

```
<xs:any namespace="##other" processContents="lax" minOccurs="0"/>
```

is added in the informative schema (see Annex B) in the elements **fuzzySystem** (after the element **anYaRuleBase**) and **knowledgeBase** (after the element **anYaDataCloud**). This extension mechanism allows the addition of new kinds of rule bases in the fuzzy system or new kinds of fuzzy variables in the knowledge base.

— Inserting of new values for attributes according to the type *customLabelType* (i.e., according to the pattern `[custom_\S*]`). This extension mechanism allows to define new labels for fuzzy operators and hedges.

Both extension methods represent a suitable approach for enabling a fast and well-defined extensibility, *conforming* to this standard.

9.1 Extensibility examples

In order to better understand the two forms of extension supported by this standard, we present two examples: the former shows how to add a new rulebase by using the wildcard-based extension mechanism; the latter shows how to add a new accumulation method by means of the definition of the type *customLabelType.*

9.1.1 Add a new rulebase

Let us consider that designers want to add a new kind of rule base named *customRuleBase*, which does not belong to the default set of rule bases provided by this standard, consisting of Mamdani rule base, TSK rule base, Tsukamoto rule base, and AnYa rule base. In this case, they shall build a custom XML schema where a new element *customRuleBase* is defined. This customised XML Schema should appear as follows:

```
<!- - custom.xsd - - >

<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified" targetNamespace="http://CUSTOMNAMESPACE"
xmlns="http://CUSTOMNAMESPACE" >
    <xs:element name="customRuleBase" type="customRuleBaseType">
        <xs:annotation>
            <xs:documentation>Custom rule base</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:complexType name="customRuleBaseType">
    ……
    </xs:complexType>
</xs:schema>
```

Once defined, the element *customRuleBase* can be used in an FML instance document as follows:

```
<!- - FMLinstance.fml - - >
<?xml version="1.0" encoding="UTF-8"?>
<fuzzySystem name="" networkAddress="" xmlns="http://www.ieee1855.org"
                                       xmlns:custom="http://CUSTOMNAMESPACE">
   <knowledgeBase networkAddress="">
   ……
   </knowledgeBase>
   <custom:customRuleBase>
   ……
   </custom:customRuleBase>
</fuzzySystem>
```

This FML instance document is *conforming* to this standard but not *strictly conforming* as described in Clause 8.

9.1.2 Add a new accumulation method

Let us consider that designers want to add a new accumulation method named *customAcc*. In this case, designers can build an FML instance document *conforming* to this standard but not *strictly conforming* as follows:

```
<!- - FMLinstance.fml - - >
<?xml version="1.0" encoding="UTF-8"?>
<fuzzySystem name="" networkAddress="" xmlns="http://www.ieee1855.org"    >
   <knowledgeBase networkAddress="">
   <fuzzyVariable name="…" domainleft="…" domainright="…" scale="" defaultValue=""
                  accumulation="custom_customAcc" defuzzifier="…" type="output">
   ……
   </fuzzyVariable>
   ……
   </knowledgeBase>
……
</fuzzySystem>
```

Annex A

(normative)

Fuzzy logic theory

When Lotfi Zadeh introduced fuzzy sets [B27] and the corresponding fuzzy logic (FL) [B26], he thought of a mathematical formalism useful for resembling human reasoning in its use of imprecise information to generate decision [B3]. Different from classical approaches, FL enables intermediate values to be defined between typical evaluations like true/false, yes/no, tall/short, fast/slow, etc. Therefore, notions such as "rather tall" or "very fast" can be formulated mathematically and handled by computers in order to apply a more human like way of thinking in the programming of computers [B28].

FL achieves this goal by "computing with words" rather than numbers. Indeed, different from mathematics that uses numerical values, FL exploits *linguistic variables* to facilitate the expression of rules and facts. Linguistic variables represent the basic block to define the fuzzy reasoning. In particular, the linguistic variables are used to form *fuzzy rules*, i.e., the main method provided by FL to capture and to simulate the human knowledge. In detail, the linguistic variables are composed of *linguistic terms*. Each linguistic term is described by a *fuzzy set*. Fuzzy sets represent an extension of conventional crisp sets. In specific, crisp sets only allow that an element fully or not at all belongs to a set, whereas, fuzzy sets enable to handle partial memberships of elements to a set. Formally, in a crisp set, membership or non-membership of an element x in a set A is described by using a function μ_A, where $\mu_A(x)=1$ if $x \in A$ and $\mu_A(x)=0$ if $x \notin A$. Instead, FL theory extends crisp set by introducing partial membership. Precisely, a fuzzy set A is featured by a membership function μ_A taking values in [0, 1]. The elements, which have been assigned the value 1, can be interpreted as the elements that are in the set A; the elements, which have been assigned the value 0, can be interpreted as the elements that are not in the set A; and the elements, which have been assigned a value between 0 and 1, can be considered as *partially* belonging to the set A. Conventionally, a fuzzy set A in a universe of discourse U may be represented by means of a set of ordered pairs, where each pair consists of a generic element x and its grade of membership $\mu_A(x)$ as follows: $A=\{(x,\mu_A(x)) \mid x \in U\}$.

In the literature, there are various types of membership functions used for defining fuzzy sets (see Figure A.1). Together with these typical membership functions, it is necessary to mention also the custom shape, i.e., a generic polygon shape built by connecting a set of points. In addition, membership functions can be defined as a logic expression of other membership functions previously defined (*circular definition*). Starting from the fuzzy set description, a linguistic variable X on a universe of discourse U is characterized by a set of linguistic terms $T_X=\{T_X^1,T_X^2,T_X^3,\ldots,T_X^N\}$, where each linguistic term T_X^i is associated with a fuzzy set that has a proper membership function defined in U. The meaning of a linguistic term can be modified through a so-called *hedge*. In this standard, the following hedges are provided:

— *Above*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The hedge *above* identifies the first point on x-axis, named x_{max}, at which *S* is a characterized by the maximum degree value. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x)=\begin{cases}0 & if\ x<x_{max}\\ 1-\mu_S(x) & if\ x\geq x_{max}\end{cases} \quad (A.1)$$

— *Any*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x)=1 \quad (A.2)$$

— *Below*, defined as follows: *L*et *S* be a fuzzy set to which the modifier must be applied. The hedge *below* identifies the first point on x-axis, named x_{max}, at which *S* is a characterized by the maximum degree value. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x) = \begin{cases} 0 & if\ x > x_{max} \\ 1 - \mu_S(x) & if\ x \le x_{max} \end{cases} \quad (A.3)$$

— *Extremely*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x) = (\mu_S(x))^3 \quad (A.4)$$

— *Intensify*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x) = \begin{cases} 2 \cdot (\mu_S(x))^2 & if\ 0 \le \mu_S(x) \le 0.5 \\ 1 - 2 \cdot (1 - \mu_S(x))^2 & if\ 0.5 < \mu_S(x) \le 1.0 \end{cases} \quad (A.5)$$

— *More_or_less*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x) = (\mu_S(x))^{\frac{1}{3}} \quad (A.6)$$

— *Norm*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied and $\mu_{\max}$ be the maximum membership degree of *S*. The hedge *norm* returns the normalized fuzzy set *M* defined as:

$$\mu_M(x) = \frac{\mu_S(x)}{\mu_{\max}} \quad (A.7)$$

— *Not*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x) = 1 - \mu_S(x) \quad (A.8)$$

— *Plus*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x) = (\mu_S(x))^{\frac{5}{4}} \quad (A.9)$$

— *Seldom*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x) = \begin{cases} \sqrt{\frac{\mu_S(x)}{2}} & \text{if } 0 \le \mu_S(x) \le 0.5 \\ 1 - \sqrt{\frac{1 - \mu_S(x)}{2}} & \text{if } 0.5 < \mu_S(x) \le 1.0 \end{cases} \quad (A.10)$$

— *Slightly*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in terms of the other hedges in the following way:

M = *intensify* [*norm* (*plus* S AND *not very* S)] (A.11)

— *Somewhat*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x) = (\mu_S(x))^{\frac{1}{2}} \tag{A.12}$$

— *Very*, defined as follows: Let *S* be a fuzzy set to which the modifier must be applied. The modified fuzzy set *M* is obtained in the following way:

$$\mu_M(x) = (\mu_S(x))^2 \tag{A.13}$$

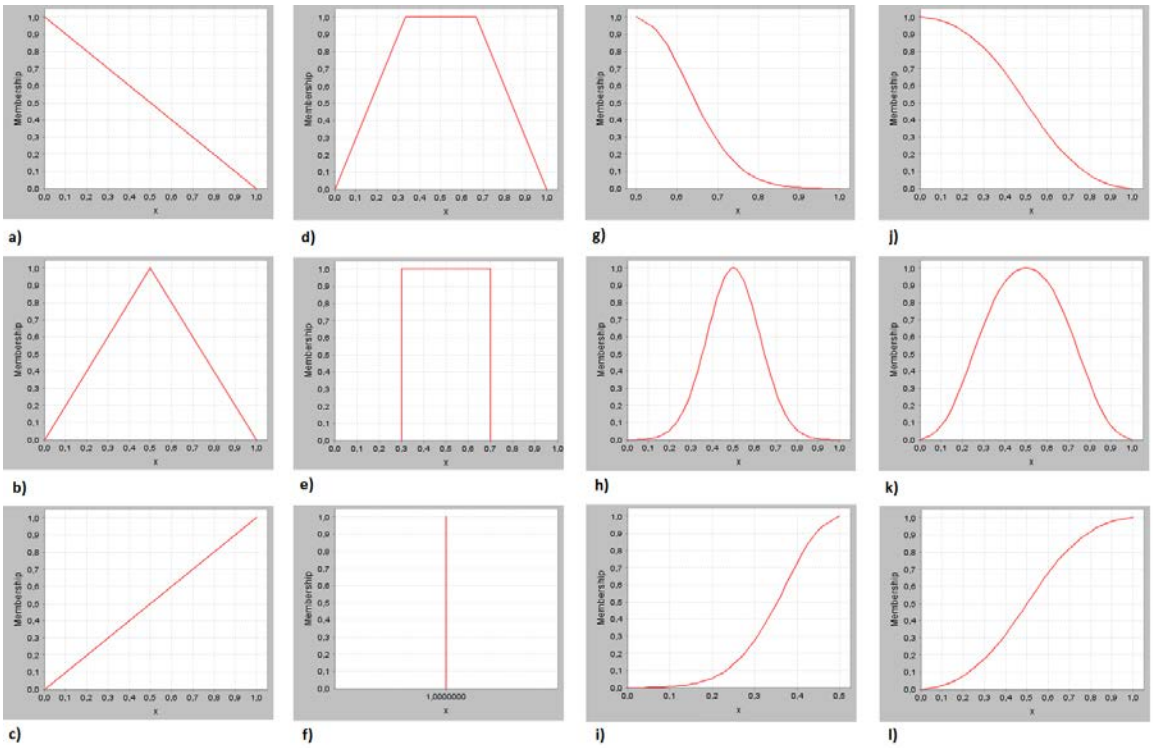


Figure A.1—Examples of membership functions: a) left linear shape,b) triangular shape, c) right linear shape, d) trapezoidal shape, e) rectangular shape, f) singleton shape, g) left Gaussian shape, h) Gaussian shape, i) right Gaussian shape, j) z-shape, k) pi shape, l) s-shape

A *fuzzy rule* can be used to define a relationship between the input and output fuzzy sets of a fuzzy inference system. A fuzzy rule assumes the form "IF *A* is *x* THEN *B* is *y* ELSE *C* is *z*", where *A*, *B*, and *C* are linguistic variables and *x*, *y*, and *z* are linguistic terms. The rule is composed of two parts: the IF part called *antecedent* and the THEN-ELSE part called *consequent*. In detail, the antecedent of the rule is "*A* is *x*," the THEN-part of the rule consequent is "*B* is *y*," and the ELSE-part of the rule consequent is "*C* is *z.*" The fuzzy relation between *A* and *B* is given by the membership function $\mu_{A\to B}(x,y) \in [0,1]$, which identifies the degree of presence or absence of association or interaction between the elements *x* and *y*. Instead, the relation between *A* and *C* is given by the membership function $\mu_{\overline{A}\to C}(x,z) \in [0,1]$. The two implications $A \to B$ and $\overline{A} \to C$ are in *or* (see [B19] for more details). Both antecedent and consequent parts of a fuzzy rule can be composed of more than one *clause* associated with fuzzy connectives such as *and* and *or.* For example, the antecedent part can be composed of two clauses in the following way "*A* is *x* and *D* is *w*" where *A* and *D* are linguistic variables, *x* and *w* are linguistic terms, and the connector is *and.* The connectors can be implemented by different operators. Using the min method (corresponding to the

intersection operation among fuzzy sets) for the connector *and* and using the max method (corresponding to the union operation among fuzzy sets) for the connector *or* are the most typical application methods for implementing the fuzzy connectives.

By using fuzzy rules, a fuzzy logic system (FLS) defines a non-linear mapping of the input data vector into - scalar outputs. A general model of a FLS is depicted in Figure A.2. As shown, an FLS contains five components: the knowledge base, the rule base, the fuzzifier, the inference engine, and the defuzzifier. The fuzzy knowledge base contains the definition of the linguistic variables used in the controlled system, corresponding to the knowledge used by human experts. The rule base represents the set of fuzzy relations among the fuzzy variables defined in the knowledge base. The Inference Engine is the fuzzy system component that is able to extract new knowledge from fuzzy knowledge base and fuzzy rule base.

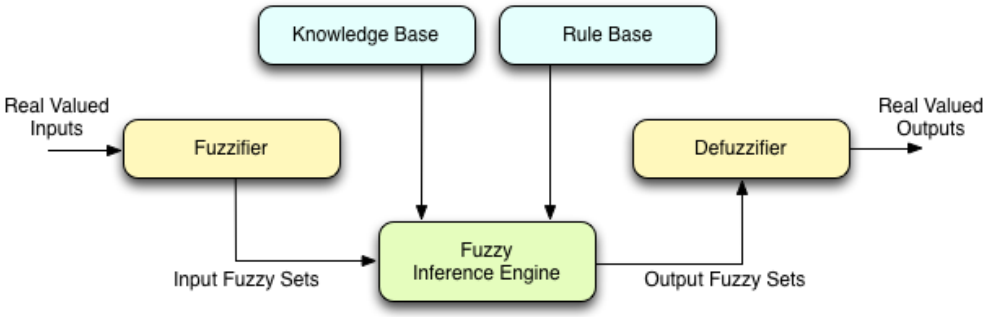


Figure A.2—FLS architecture [B1]

Precisely, the fuzzy inference consists of the following three steps: aggregation, activation, and accumulation. The aggregation step performs the combination of the multiple clauses contained in the antecedent part of a rule by means two connectors, *and* or *or*, implemented respectively, through *t-norm* and *t-conorm* operators. The following *t-norm* operators are already provided in this standard:

— *Minimum t-norm* (MIN) *T* is formally defined as follows:

$$T(a,b) = \min(a,b) \tag{A.14}$$

— *Product t-norm* (PROD) *T* is formally defined as follows:

$$T(a,b) = a \cdot b \tag{A.15}$$

— *Łukasiewicz t-norm* or *Bounded difference* (BDIF) *T* is formally defined as follows:

$$T(a,b) = \max(0, a+b-1) \tag{A.16}$$

— *Drastic t-norm* or *drastic product* (DRP) *T* is formally defined as follows:

$$T(a,b) = \begin{cases} b & if\ a = 1 \\ a & if\ b = 1 \\ 0 & otherwise \end{cases} \tag{A.17}$$

— *Einstein product* (EPROD) *T* is formally defined as follows:

$$T(a,b) = \frac{a \cdot b}{2-(a+b-a \cdot b)} \tag{A.18}$$

— *Hamacher product* (HPROD) *T* is formally defined as follows:

$$T(a,b) = \frac{a+b}{a+b-a\cdot b} \tag{A.19}$$

— *Nilpotent minum* (NILMIN) *T* is formally defined as follows:

$$T(a,b) = \begin{cases} \min(a,b) & if\ a+b>1 \\ 0 & otherwise \end{cases} \tag{A.20}$$

The following *t-conorm* operators are already provided in this standard:

— *Maximum t-conorm* (MAX) *T** is formally defined as follows:

$$T^*(a,b) = \max(a,b) \tag{A.21}$$

— *Probabilistc sum* (PROBOR) *T** is formally defined as follows:

$$T^*(a,b) = a+b-a\cdot b \tag{A.22}$$

— *Bounded sum* (BSUM) *T** is formally defined as follows:

$$T^*(a,b) = \min(1, a+b) \tag{A.23}$$

— *Drastic t-conorm* or *drastic sum* (DRS) *T** is formally defined as follows:

$$T^*(a,b) = \begin{cases} b & if\ a=0 \\ a & if\ b=0 \\ 1 & otherwise \end{cases} \tag{A.24}$$

— *Einstein sum* (ESUM) *T** is formally defined as follows:

$$T^*(a,b) = \frac{a+b}{1+a\cdot b} \tag{A.25}$$

— *Hamacher sum* (HSUM) *T** is formally defined as follows:

$$T^*(a,b) = \frac{a+b-2\cdot a\cdot b}{1-a\cdot b} \tag{A.26}$$

— *Nilpotent maximum* (NILMAX) *T** is formally defined as follows:

$$T^*(a,b) = \begin{cases} \max(a,b) & if\ a+b<1 \\ 1 & otherwise \end{cases} \tag{A.27}$$

The activation step is devoted to determine the degree of membership of the consequent part according to the degree of accomplishment of the antecedent part of a rule. In other words, the activation step is responsible for executing the rule implication. In literature, there are different implication methods. In this standard, the following implication methods are natively provided:

— *Mamdani implication* (MIN) is formally defined as follows:

$$\mu_{A\to B}(x,y) = \min\{\mu_A(x), \mu_B(y)\} \tag{A.28}$$

— *Larsen Implication* (PROD) is formally defined as follows:

$$\mu_{A\to B}(x,y) = \mu_A(x)\cdot\mu_B(y) \tag{A.29}$$

— *Bounded difference Implication* (BDIF) is formally defined as follows:

$$\mu_{A\to B}(x,y) = \max(0,\mu_A(x)+\mu_B(y)-1) \tag{A.30}$$

— *Drastic product Implication* (DRP) is formally defined as follows:

$$\mu_{A\to B}(x,y) = \begin{cases} \mu_A(x) & if\ \mu_B(y)=1 \\ \mu_B(y) & if\ \mu_A(x)=1 \\ 0 & otherwise \end{cases} \tag{A.31}$$

— *Einstein product Implication* (EPROD) is formally defined as follows:

$$\mu_{A\to B}(x,y) = \frac{\mu_A(x)\cdot\mu_B(y)}{2-(\mu_A(x)+\mu_B(y)-\mu_A(x)\cdot\mu_B(y))} \tag{A.32}$$

— *Hamacher product Implication* (HPROD) is formally defined as follows:

$$\mu_{A\to B}(x,y) = \frac{\mu_A(x)+\mu_B(y)}{\mu_A(x)+\mu_B(y)-\mu_A(x)\cdot\mu_B(y)} \tag{A.33}$$

— *Nilpotent minum Implication* (NILMIN) *T* is formally defined as follows:

$$\mu_{A\to B}(x,y) = \begin{cases} \min(\mu_A(x),\mu_B(y)) & if\ \mu_A(x)+\mu_B(y)>1 \\ 0 & otherwise \end{cases} \tag{A.34}$$

Finally, the accumulation step aims to combine the fuzzy sets representing the results of all rules in a rule base to obtain the overall fuzzy set. In this standard, the following accumulation methods are natively provided:

— *Maximum* (MAX) consists in performing the maximum operation among the fuzzy sets representing the outputs of the rules. Let *F1*, *F2*, … , *FN* be the fuzzy sets resulting from the *N* rules and *U* be the universe of the discourse, the fuzzy set *S* resulting from the maximum operation is defined as follows:

$$\mu_S(x) = \max\{\mu_{F1}(x),\mu_{F2}(x),\ldots,\mu_{FN}(x)\}\ \ \forall x\in U \tag{A.35}$$

— *Probabilistic sum* (PROBOR) consists in performing the probor operation among the fuzzy sets representing the outputs of the rules. Let *F1*, *F2*, … , *FN* be the fuzzy sets resulting from the *N* rules and *U* be the universe of the discourse, the fuzzy set *S* resulting from the probor operation is defined as follows:

$$\mu_S(x) = \mu_{F1}(x)+\mu_{F2}(x)+\ldots+\mu_{FN}(x)-\mu_{F1}(x)\cdot\mu_{F2}(x)\cdot\ldots\cdot\mu_{FN}(x)\ \ \forall x\in U \tag{A.36}$$

— *Bounded Sum* (BSUM) consists in performing the bounded sum operation among the fuzzy sets representing the outputs of the rules. Let *F1*, *F2*, … , *FN* be the fuzzy sets resulting from the *N* rules and *U* be the universe of the discourse, the fuzzy set *S* resulting from the bounded sum operation is defined as follows:

$$\mu_S(x) = \min\{1,(\mu_{F1}(x)+\mu_{F2}(x)+\ldots+\mu_{FN}(x))\}\ \ \forall x\in U \tag{A.37}$$

— *Drastic drastic sum* (DRS) consists in performing the drastic sum operation among the fuzzy sets representing the outputs of the rules. Let F_1, F_2, … , F_N be the fuzzy sets resulting from the *N* rules

and *U* be the universe of the discourse, the fuzzy set *S* resulting from the drustic sum operation is defined as follows:

$$\mu_S(x) = \begin{cases} \mu_{F_1}(x) & \text{if } \mu_{F_i}(x) = 0 \ \ \forall i \in \{1,2,\cdots,N\} \text{ and } i \neq 1 \\ \mu_{F_2}(x) & \text{if } \mu_{F_i}(x) = 0 \ \ \forall i \in \{1,2,\cdots,N\} \text{ and } i \neq 2 \\ \vdots & \\ \mu_{F_N}(x) & \text{if } \mu_{F_i}(x) = 0 \ \ \forall i \in \{1,2,\cdots,N\} \text{ and } i \neq N \\ 1 & \text{otherwise} \end{cases} \quad \forall x \in U \tag{A.38}$$

— *Einstein sum* (ESUM) consists in performing the Einstein sum operation among the fuzzy sets representing the outputs of the rules. Let *F1*, *F2*, …, *FN* be the fuzzy sets resulting from the *N* rules and *U* be the universe of the discourse, the fuzzy set *S* resulting from the Einsten sum operation is defined as follows:

$$\mu_S(x) = \frac{\mu_{F1}(x) + \mu_{F2}(x) + \ldots + \mu_{FN}(x)}{1 + \mu_{F1}(x) \cdot \mu_{F2}(x) \cdot \ldots \cdot \mu_{FN}(x)} \ \ \forall x \in U \tag{A.39}$$

— *Hamacher sum* (HSUM) consists in performing the Hamacher sum operation among the fuzzy sets representing the outputs of the rules. Let *F1*, *F2*, … , *FN* be the fuzzy sets resulting from the *N* rules and *U* be the universe of the discourse, the fuzzy set *S* resulting from the Hamacher sum operation is defined as follows:

$$\mu_S(x) = \frac{\mu_{F1}(x) + \mu_{F2}(x) + \ldots + \mu_{FN}(x) - 2 \cdot \mu_{F1}(x) \cdot \mu_{F2}(x) \cdot \ldots \cdot \mu_{FN}(x)}{1 - \mu_{F1}(x) \cdot \mu_{F2}(x) \cdot \ldots \cdot \mu_{FN}(x)} \ \ \forall x \in U \tag{A.40}$$

— *Nilpotent maximum* (NILMAX) consists in performing the Nilpotent maximum operation among the fuzzy sets representing the outputs of the rules. Let *F1*, *F2*, … , *FN* be the fuzzy sets resulting from the *N* rules and *U* be the universe of the discourse, the fuzzy set *S* resulting from the Nilpotent maximum operation is defined as follows:

$$\mu_S(x) = \begin{cases} \max(\mu_{F1}(x), \mu_{F2}(x), \ldots, \mu_{FN}(x) & \text{if } \mu_{F1}(x) + \mu_{F2}(x) + \ldots + \mu_{FN}(x) < 1 \\ 1 & \text{otherwise} \end{cases} \quad \forall x \in U \tag{A.41}$$

In conclusion, the controlled system works with real numbers, whereas the fuzzy system works with fuzzy variables. The two modules, the fuzzification and the defuzzification, are necessary to bridge the controlled systems with the fuzzy systems. The former permits to transform the real numbers used by the controlled systems into fuzzy sets used by the fuzzy system. The latter transforms the fuzzy set generated by the fuzzy system into crisp values usable by the controlled system. In the literature, there are different defuzzification methods. The following defuzzification methods are already provided in this standard:

— *Mean of Maxima* (MOM) consists of choosing the mean of those points where the membership function is at a maximum. Let *A* be the fuzzy set obtained by the accumulation phase and μ_{max} be its maximum membership degree; the defuzzification value *u* is formally defined as follows:

$$u = \frac{\sum_{x \in S} x}{|S|} \tag{A.42}$$

where $S = \{x \mid \mu_A(x) = \mu_{max}\}$.

— *Leftmost Maximum* (LM) consists in choosing the smallest point that have maximum membership. Let *A* be the fuzzy set obtained by the accumulation phase and μ_{max} be its maximum membership degree; the defuzzification value *u* is formally defined as follows:

$$u = \min S \tag{A.43}$$

where $S = \{x \mid \mu_A(x) = \mu_{max}\}$.

— *Rightmost Maximum* (RM) consists in choosing the largest point that have maximum membership. Let *A* be the fuzzy set obtained by the accumulation phase and μ_{max} be its maximum membership degree; the defuzzification value *u* is formally defined as follows:

$$u = \max S \tag{A.44}$$

where $S = \{x \mid \mu_A(x) = \mu_{max}\}$.

— *Center of Gravity* (COG) consists in choosing the abscissa under the centre of gravity of the fuzzy set. Let *A* be the fuzzy set obtained by the accumulation phase, *U* be the universe of the discourse of the fuzzy set *A*; the defuzzification value *u* is formally defined as follows:

$$u = \frac{\int_U x \cdot \mu_A(x) dx}{\int_U \mu_A(x) dx} \tag{A.45}$$

where $\mu_A(x)$ is the membership degree of the point *x* to the fuzzy set *A*.

— *Center of Area* (COA) consists in choosing the value that has one half of the area of the fuzzy set on each side of it. Let *A* be the fuzzy set obtained by the accumulation phase; the defuzzification value *u* is formally defined as follows:

$$u = \{x \mid \int_{Min}^{x} \mu_A(x) dx = \int_{x}^{Max} \mu_A(x) dx\} \tag{A.46}$$

where *Min* and *Max* are the left side and the right side of the universe of the discourse of the fuzzy set *A*, respectively.

In literature, there are also different kinds of FLSs. The best-known ones are the Mamdani system [B16] and the Takagi-Sugeno-Kang (TSK) system [B20]. They differ for the nature of the consequent part of fuzzy rules in the rule base component. Indeed, the Mamdani system uses fuzzy sets to model the consequent part of a rule, whereas the TSK system uses a linear function of input variables. As a consequence, the rule outputs are not fuzzy sets but crisp values that must be combined through an aggregation process. In this standard, the typical weighted average (WA) method is already provided. Formally, the WA method computes the overall output *u* as follows:

$$u = \frac{\sum_{i=1}^{N} z_i \cdot w_i}{\sum_{i=1}^{N} w_i} \tag{A.47}$$

where *N* is the number of rules, z_i is the output of the *i*-th rule, and the w_i is the firing strength of the *i*-th rule.

Recently, new fuzzy systems have been developed. One of the most known is called AnYa (Angelog-Yager) [B5]. Its main feature is being free of logical connectives and membership functions in the antecedent part of the fuzzy rule. Indeed, unlike Mamdani and TSK systems, in which the antecedent part (IF) is represented by a fuzzy set, in AnYa system the antecedent part relates the analyzed information to a data set with no pre-defined shape, called Data Cloud. As for the consequent part of a rule in the AnYa system, it can be the same as in Mamdani or TSK.

Annex B

(normative)

XML Schema

FML.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified"
attributeFormDefault="unqualified" targetNamespace="http://www.ieee1855.org"
xmlns="http://www.ieee1855.org" >
    <xs:element name="fuzzySystem" type="fuzzySystemType">
        <xs:annotation>
            <xs:documentation>Fuzzy System</xs:documentation>
        </xs:annotation>
    </xs:element>
    <xs:complexType name="fuzzySystemType">
        <xs:sequence>
            <xs:element name="knowledgeBase" type="knowledgeBaseType">
                <xs:annotation>
                    <xs:documentation>Fuzzy Concepts Collection</xs:documentation>
                </xs:annotation>
            </xs:element>
            <xs:choice minOccurs="1" maxOccurs="unbounded">
                <xs:element name="mamdaniRuleBase" type="ruleBaseType">
                    <xs:annotation>
                        <xs:documentation>Mamdani Fuzzy Rules
                                          Collection</xs:documentation>
                    </xs:annotation>
                </xs:element>
                <xs:element name="tsukamotoRuleBase" type="ruleBaseType">
                    <xs:annotation>
                        <xs:documentation>Tsukamoto Fuzzy Rules
                                          Collection</xs:documentation>
                    </xs:annotation>
                </xs:element>
                <xs:element name="tskRuleBase" type="tskRuleBaseType">
                    <xs:annotation>
                        <xs:documentation>TSK Fuzzy Rules Collection</xs:documentation>
                    </xs:annotation>
                </xs:element>
                <xs:element name="anYaRuleBase" type="anYaRuleBaseType">
                    <xs:annotation>
                        <xs:documentation>AnYa Fuzzy Rules Collection</xs:documentation>
                    </xs:annotation>
                </xs:element>
                <xs:any namespace="##other" processContents="lax" minOccurs="0"/>
            </xs:choice>
        </xs:sequence>
        <xs:attribute name="name" type="xs:ID" use="required"/>
        <xs:attribute name="networkAddress" default="127.0.0.1"
                      type="networkAddressType"/>
    </xs:complexType>
    <xs:complexType name="knowledgeBaseType">
        <xs:sequence>
            <xs:choice minOccurs="2" maxOccurs="unbounded">
             <xs:element name="fuzzyVariable" type="fuzzyVariableType">
                <xs:annotation>
                    <xs:documentation>Fuzzy Concept Item composed of fuzzy
                                      sets</xs:documentation>
                </xs:annotation>
             </xs:element>
             <xs:element name="tsukamotoVariable" type="tsukamotoVariableType">
```

```
            <xs:annotation>
                <xs:documentation>Fuzzy Concept Item composed of fuzzy
                                  sets</xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="aggregatedFuzzyVariable"
                      type="aggregatedFuzzyVariableType">
            <xs:annotation>
                <xs:documentation>Linear function coefficients</xs:documentation>
            </xs:annotation>
          </xs:element>
          <xs:element name="tskVariable" type="tskVariableType">
            <xs:annotation>
                <xs:documentation>Linear function coefficients</xs:documentation>
            </xs:annotation>
        </xs:element>
          <xs:element name="anYaDataCloud" type="anYaDataCloudType">
            <xs:annotation>
                <xs:documentation>Data cloud to be used in the rule antecedent part
                                  of a cloud-based fuzzy system</xs:documentation>
            </xs:annotation>
        </xs:element>
          <xs:any namespace="##other" processContents="lax" minOccurs="0"/>
       </xs:choice>
    </xs:sequence>
     <xs:attribute name="networkAddress" default="127.0.0.1"
                   type="networkAddressType"/>
</xs:complexType>
<xs:complexType name="fuzzyVariableType">
    <xs:sequence>
        <xs:element name="fuzzyTerm" type="fuzzyTermType" maxOccurs="unbounded">
            <xs:annotation>
                <xs:documentation>Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="name" type="xs:ID" use="required"/>
    <xs:attribute name="scale" type="xs:string" />
    <xs:attribute name="domainleft" type="xs:float"  use="required"/>
    <xs:attribute name="domainright" type="xs:float" use="required"/>
    <xs:attribute name="type" type="typeType" default="input"/>
    <xs:attribute name="accumulation" type="accumulationType" default="MAX" />
    <xs:attribute name="defuzzifier" type="defuzzifierType" default="COG" />
    <xs:attribute name="defaultValue" type="xs:float" default="0"/>
    <xs:attribute name="networkAddress" default="127.0.0.1"
                  type="networkAddressType"/>
</xs:complexType>
<xs:complexType name="fuzzyTermType">
    <xs:choice>
        <xs:element name="rightLinearShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Right Linear Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="leftLinearShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Left Linear Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="piShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Pishape Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="triangularShape" type="threeParamType" >
            <xs:annotation>
                <xs:documentation>Triangle Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="gaussianShape" type="twoParamType" >
            <xs:annotation>
```

```
                <xs:documentation>Gaussian Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="rightGaussianShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Right Gaussian Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="leftGaussianShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Left Gaussian Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="trapezoidShape" type="fourParamType" >
            <xs:annotation>
                <xs:documentation>Trapezoid Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="singletonShape" type="oneParamType" >
            <xs:annotation>
                <xs:documentation>Singleton Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="rectangularShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Rectagle Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="zShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Z Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="sShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>S Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="pointSetShape" type="pointSetShapeType" >
            <xs:annotation>
                <xs:documentation>Fuzzy Set based on a set of
                                points</xs:documentation>
            </xs:annotation>
        </xs:element>
          <xs:element name="circularDefinition" type="circularDefinitionType" >
            <xs:annotation>
                <xs:documentation>Fuzzy Set based on a circular
                                definition</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="customShape" type="customShapeType" >
            <xs:annotation>
                <xs:documentation>Custom Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:choice>
    <xs:attribute name="name" type="xs:ID" use="required"/>
    <xs:attribute name="complement" default="false">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:pattern value="true|false|TRUE|FALSE|True|False"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>
<xs:complexType name="tsukamotoVariableType">
    <xs:sequence>
        <xs:element name="tsukamotoTerm" type="tsukamotoTermType"
                    maxOccurs="unbounded">
            <xs:annotation>
                <xs:documentation>Fuzzy Set</xs:documentation>
```

```
            </xs:annotation>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="name" type="xs:ID" use="required"/>
    <xs:attribute name="scale" type="xs:string" />
    <xs:attribute name="domainleft" type="xs:float"  use="required"/>
    <xs:attribute name="domainright" type="xs:float" use="required"/>
    <xs:attribute name="type" type="xs:string" fixed="output" use="required"/>
    <xs:attribute name="combination" type="combinationType" default="WA"/>
    <xs:attribute name="defaultValue" type="xs:float" default="0"/>
    <xs:attribute name="networkAddress" default="127.0.0.1"
                  type="networkAddressType"/>
</xs:complexType>
<xs:complexType name="tsukamotoTermType">
    <xs:choice>
        <xs:element name="rightLinearShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Right Linear Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="leftLinearShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Left Linear Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="rightGaussianShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Right Gaussian Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="leftGaussianShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Left Gaussian Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="zShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>Z Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="sShape" type="twoParamType" >
            <xs:annotation>
                <xs:documentation>S Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="pointSetMonotonicShape" type="pointSetMonotonicShapeType" >
            <xs:annotation>
                <xs:documentation>Fuzzy Set based on a set of
                                  points</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="customMonotonicShape" type="customShapeType" >
            <xs:annotation>
                <xs:documentation>Custom Fuzzy Set</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:choice>
    <xs:attribute name="name" type="xs:ID" use="required"/>
    <xs:attribute name="complement" default="false">
        <xs:simpleType>
            <xs:restriction base="xs:string">
                <xs:pattern value="true|false|TRUE|FALSE|True|False"/>
            </xs:restriction>
        </xs:simpleType>
    </xs:attribute>
</xs:complexType>
<xs:complexType name="oneParamType">
    <xs:attribute name="param1" type="xs:float" use="required"/>
</xs:complexType>
<xs:complexType name="twoParamType">
    <xs:attribute name="param1" type="xs:float" use="required"/>
```

```
    <xs:attribute name="param2" type="xs:float" use="required"/>
</xs:complexType>
<xs:complexType name="threeParamType">
    <xs:attribute name="param1" type="xs:float" use="required"/>
    <xs:attribute name="param2" type="xs:float" use="required"/>
    <xs:attribute name="param3" type="xs:float" use="required"/>
</xs:complexType>
<xs:complexType name="fourParamType">
    <xs:attribute name="param1" type="xs:float" use="required"/>
    <xs:attribute name="param2" type="xs:float" use="required"/>
    <xs:attribute name="param3" type="xs:float" use="required"/>
    <xs:attribute name="param4" type="xs:float" use="required"/>
</xs:complexType>
<xs:complexType name="pointSetShapeType">
    <xs:sequence>
        <xs:element name="point" type="pointType" minOccurs="1"
                    maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="interpolationMethod" type="interpolationMethodType"
                  default="linear"/>
    <xs:attribute name="degree" type="xs:int" default="3"/>
</xs:complexType>
<xs:complexType name="pointSetMonotonicShapeType">
    <xs:sequence>
        <xs:element name="point" type="pointType" minOccurs="1"
                    maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="interpolationMethod" type="monotonicInterpolationMethodType"
                  default="linear"/>
</xs:complexType>
<xs:complexType name="pointType">
    <xs:attribute name="x" type="xs:float" use="required"/>
    <xs:attribute name="y" type="xs:float" use="required"/>
</xs:complexType>
 <xs:complexType name="customShapeType">
    <xs:sequence>
      <xs:element name="parameter" type="parameterType" minOccurs="0"
                  maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="name" type="customLabelType" use="required"/>
</xs:complexType>
<xs:complexType name="parameterType">
    <xs:attribute name="value" type="xs:string" use="required"/>
</xs:complexType>
<xs:complexType name="circularDefinitionType">
    <xs:choice>
        <xs:element name="and" type="andLogicalType"/>
        <xs:element name="or" type="orLogicalType"/>
    </xs:choice>
</xs:complexType>
<xs:complexType name="andLogicalType">
    <xs:choice>
                  <xs:sequence>
                       <xs:element name="termName" type="circularTermType"
                                   minOccurs="2" maxOccurs="2" />
                  </xs:sequence>
                  <xs:sequence>
                       <xs:choice>
                           <xs:element name="and" type="andLogicalType"/>
                           <xs:element name="or" type="orLogicalType"/>
                       </xs:choice>
                       <xs:element name="termName" type="circularTermType"/>
                  </xs:sequence>
    </xs:choice>
    <xs:attribute name="t-norm" type="andMethodType" default="MIN"/>
</xs:complexType>
 <xs:complexType name="orLogicalType">
    <xs:choice>
                  <xs:sequence>
                       <xs:element name="termName" type="circularTermType"
                                   minOccurs="2" maxOccurs="2" />
```

```
                    </xs:sequence>
                    <xs:sequence>
                         <xs:choice>
                             <xs:element name="and" type="andLogicalType"/>
                             <xs:element name="or" type="orLogicalType"/>
                         </xs:choice>
                         <xs:element name="termName" type="circularTermType" />
                    </xs:sequence>
      </xs:choice>
      <xs:attribute name="t-conorm" type="orMethodType" default="MAX" />
  </xs:complexType>
  <xs:complexType name="circularTermType">
   <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="complement" default="false">
          <xs:simpleType>
              <xs:restriction base="xs:string">
                  <xs:pattern value="true|false|TRUE|FALSE|True|False"/>
              </xs:restriction>
          </xs:simpleType>
      </xs:attribute>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="aggregatedFuzzyVariableType">
      <xs:sequence>
          <xs:element name="aggregatedFuzzyTerm" type="aggregatedFuzzyTermType"
                      maxOccurs="unbounded">
              <xs:annotation>
                  <xs:documentation>Aggregated Fuzzy Set</xs:documentation>
              </xs:annotation>
          </xs:element>
      </xs:sequence>
      <xs:attribute name="name" type="xs:ID" use="required"/>
      <xs:attribute name="type" type="typeType" fixed="input"/>
      <xs:attribute name="networkAddress" default="127.0.0.1"
                    type="networkAddressType"/>
</xs:complexType>
<xs:complexType name="aggregatedFuzzyTermType">
      <xs:choice>
          <xs:element name="and" type="andAggregatedType"/>
          <xs:element name="or" type="orAggregatedType"/>
      </xs:choice>
      <xs:attribute name="name" type="xs:ID" use="required"/>
</xs:complexType>
<xs:complexType name="andAggregatedType">
      <xs:choice>
                    <xs:sequence>
                         <xs:element name="clause" type="clauseType"  minOccurs="2"
                                     maxOccurs="2" />
                    </xs:sequence>
                    <xs:sequence>
                         <xs:choice>
                             <xs:element name="and" type="andAggregatedType"/>
                             <xs:element name="or" type="orAggregatedType"/>
                         </xs:choice>
                         <xs:element name="clause" type="clauseType"/>
                    </xs:sequence>
      </xs:choice>
      <xs:attribute name="t-norm" type="andMethodType" default="MIN"/>
  </xs:complexType>
   <xs:complexType name="orAggregatedType">
      <xs:choice>
                    <xs:sequence>
                         <xs:element name="clause" type="clauseType"  minOccurs="2"
                                     maxOccurs="2" />
                    </xs:sequence>
                    <xs:sequence>
                         <xs:choice>
                             <xs:element name="and" type="andAggregatedType"/>
                             <xs:element name="or" type="orAggregatedType"/>
```

```
                        </xs:choice>
                        <xs:element name="clause" type="clauseType" />
                    </xs:sequence>
        </xs:choice>
        <xs:attribute name="t-conorm" type="orMethodType" default="MAX" />
</xs:complexType>
<xs:complexType name="tskVariableType">
        <xs:sequence>
            <xs:element name="tskTerm" type="tskTermType"  maxOccurs="unbounded">
                <xs:annotation>
                    <xs:documentation>TSK Term</xs:documentation>
                </xs:annotation>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="name" type="xs:ID" use="required"/>
        <xs:attribute name="scale" type="xs:string"/>
        <xs:attribute name="type" type="xs:string" fixed="output" use="required"/>
        <xs:attribute name="combination" type="combinationType" default="WA"/>
        <xs:attribute name="defaultValue" type="xs:float" default="0"/>
        <xs:attribute name="networkAddress" default="127.0.0.1"
                      type="networkAddressType"/>
    </xs:complexType>
    <xs:complexType name="tskTermType">
        <xs:sequence>
              <xs:element name="tskValue" type="xs:float" maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="name" type="xs:ID" use="required"/>
        <xs:attribute name="order" type="tskOrderType" use="required"/>
    </xs:complexType>
     <xs:complexType name="anYaDataCloudType">
        <xs:sequence>
            <xs:element name="datum" type="xs:double" minOccurs="1"
                        maxOccurs="unbounded"/>
        </xs:sequence>
        <xs:attribute name="name" type="xs:ID" use="required"/>
        <xs:attribute name="networkAddress" default="127.0.0.1"
                      type="networkAddressType"/>
    </xs:complexType>
    <xs:complexType name="ruleBaseType">
         <xs:sequence>
            <xs:element name="rule" type="fuzzyRuleType" minOccurs="1"
                        axOccurs="unbounded">
                  <xs:annotation>
                       <xs:documentation>Rule in a Mamdani system</xs:documentation>
                  </xs:annotation>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="name" type="xs:ID" use="required"/>
        <xs:attribute name="activationMethod" type="activationMethodType" default="MIN"/>
        <xs:attribute name="andMethod" type="andMethodType" default="MIN"/>
        <xs:attribute name="orMethod" type="orMethodType" default="MAX" />
        <xs:attribute name="networkAddress" default="127.0.0.1"
                      type="networkAddressType"/>
    </xs:complexType>
    <xs:complexType name="tskRuleBaseType">
      <xs:sequence>
            <xs:element name="tskRule" type="tskFuzzyRuleType" minOccurs="1"
                        maxOccurs="unbounded">
                  <xs:annotation>
                       <xs:documentation>Rule in a TSK system</xs:documentation>
                  </xs:annotation>
            </xs:element>
        </xs:sequence>
        <xs:attribute name="name" type="xs:ID" use="required"/>
        <xs:attribute name="activationMethod" type="activationMethodType" default="MIN"/>
        <xs:attribute name="andMethod" type="andMethodType" default="MIN"/>
        <xs:attribute name="orMethod" type="orMethodType" default="MAX" />
        <xs:attribute name="networkAddress" default="127.0.0.1"
                      type="networkAddressType"/>
    </xs:complexType>
    <xs:complexType name="anYaRuleBaseType">
```

```
    <xs:sequence>
        <xs:element name="anYaRule" type="anYaRuleType" minOccurs="1"
                    maxOccurs="unbounded">
            <xs:annotation>
                <xs:documentation>Rule in a cloud-based fuzzy
                                  system</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="name" type="xs:ID" use="required"/>
    <xs:attribute name="activationMethod" type="activationMethodType" default="MIN"/>
    <xs:attribute name="networkAddress" default="127.0.0.1"
                  type="networkAddressType"/>
</xs:complexType>
<xs:complexType name="fuzzyRuleType">
    <xs:sequence>
        <xs:element name="antecedent" type="antecedentType">
            <xs:annotation>
                <xs:documentation>IF-part of a fuzzy rule</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="consequent" type="consequentType">
           <xs:annotation>
                <xs:documentation>THEN[-ELSE]-part of a fuzzy rule</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="name" type="xs:ID" use="required"/>
    <xs:attribute name="andMethod" type="andMethodType" default="MIN"/>
    <xs:attribute name="orMethod" type="orMethodType" default="MAX" />
    <xs:attribute name="connector" type="connectorType" default="and"/>
    <xs:attribute name="weight" type="weightType" default="1.0"/>
    <xs:attribute name="networkAddress" default="127.0.0.1"
                  type="networkAddressType"/>
</xs:complexType>
<xs:complexType name="tskFuzzyRuleType">
    <xs:sequence>
        <xs:element name="antecedent" type="antecedentType">
            <xs:annotation>
                <xs:documentation>IF-part of a rule</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="tskConsequent" type="tskConsequentType">
            <xs:annotation>
                <xs:documentation>THEN[-ELSE}-part of a TSK rule</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="name" type="xs:ID" use="required"/>
    <xs:attribute name="andMethod" type="andMethodType" default="MIN"/>
    <xs:attribute name="orMethod" type="orMethodType" default="MAX" />
    <xs:attribute name="connector" type="connectorType" default="and"/>
    <xs:attribute name="weight" type="weightType" default="1.0"/>
    <xs:attribute name="networkAddress" default="127.0.0.1"
                  type="networkAddressType"/>
</xs:complexType>
<xs:complexType name="anYaRuleType">
    <xs:sequence>
        <xs:element name="anYaAntecedent" type="anYaAntecedentType">
            <xs:annotation>
                <xs:documentation>IF-part of a fuzzy rule in a cloud-based fuzzy
                                  system</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:choice>
                        <xs:element name="consequent" type="consequentType">
                          <xs:annotation>
                              <xs:documentation>THEN[-ELSE]-part of a Mamdani
                                                rule</xs:documentation>
                          </xs:annotation>
                        </xs:element>
```

```
                        <xs:element name="tskConsequent" type="tskConsequentType">
                          <xs:annotation>
                              <xs:documentation>THEN[-ELSE]-part of a TSK fuzzy
                                                rule</xs:documentation>
                          </xs:annotation>
                        </xs:element>
      </xs:choice>
    </xs:sequence>
    <xs:attribute name="name" type="xs:ID" use="required"/>
    <xs:attribute name="weight" type="weightType" default="1.0"/>
    <xs:attribute name="networkAddress" default="127.0.0.1"
                  type="networkAddressType"/>
</xs:complexType>
<xs:complexType name="antecedentType">
    <xs:sequence>
        <xs:element name="clause" type="clauseType" maxOccurs="unbounded">
            <xs:annotation>
                <xs:documentation>Antecedent Clause Fuzzy Rule</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="anYaAntecedentType">
    <xs:sequence>
        <xs:element name="dataCloudName" type="xs:IDREF" >
            <xs:annotation>
                <xs:documentation>Antecedent Clause Fuzzy Rule in a cloud-based fuzzy
                                  system</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="consequentType">
    <xs:sequence>
        <xs:element name="then" type="consequentClausesType">
            <xs:annotation>
                <xs:documentation>THEN-part of a fuzzy rule</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="else" type="consequentClausesType" minOccurs="0">
            <xs:annotation>
                <xs:documentation>ELSE-part of a fuzzy rule</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="consequentClausesType">
    <xs:sequence>
        <xs:element name="clause" type="clauseType" maxOccurs="unbounded">
            <xs:annotation>
                <xs:documentation>Consequent Mamdani Clause Fuzzy
                                  Rule</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="tskConsequentType">
    <xs:sequence>
        <xs:element name="tskThen" type="tskConsequentClausesType">
            <xs:annotation>
                <xs:documentation>THEN-part of a TSK fuzzy rule</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="tskElse" type="tskConsequentClausesType" minOccurs="0">
            <xs:annotation>
                <xs:documentation>ELSE-part of a TSK fuzzy rule</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="tskConsequentClausesType">
```

```
    <xs:sequence>
        <xs:element name="tskClause" type="tskClauseType" maxOccurs="unbounded">
            <xs:annotation>
                <xs:documentation>Consequent Tsk Clause Fuzzy Rule</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="clauseType">
    <xs:sequence>
        <xs:element name="variable" type="xs:IDREF">
            <xs:annotation>
                <xs:documentation>Clause Fuzzy Variable</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="term" type="xs:IDREF">
            <xs:annotation>
                <xs:documentation>Clause Fuzzy Term</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
    <xs:attribute name="modifier" use="optional" type="modifierType"/>
</xs:complexType>
<xs:complexType name="tskClauseType">
    <xs:sequence>
        <xs:element name="variable" type="xs:IDREF">
            <xs:annotation>
                <xs:documentation>Clause Fuzzy Variable</xs:documentation>
            </xs:annotation>
        </xs:element>
        <xs:element name="term" type="xs:IDREF">
            <xs:annotation>
                <xs:documentation>Clause Fuzzy Term</xs:documentation>
            </xs:annotation>
        </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:simpleType name="standardDefuzzifierType">
            <xs:restriction base="xs:string">
                <!-- MOM reference: Equation (A.42) from IEEE Std 1855 -->
                <xs:enumeration value="MOM"></xs:enumeration>
                <!-- LM reference: Equation (A.43) from IEEE Std 1855 -->
                <xs:enumeration value="LM"></xs:enumeration>
                <!-- RM reference: Equation (A.44) from IEEE Std 1855 -->
                <xs:enumeration value="RM"></xs:enumeration>
                <!-- COG reference: Equation (A.45) from IEEE Std 1855 -->
                <xs:enumeration value="COG"></xs:enumeration>
                <!-- COA reference: Equation (A.46) from IEEE Std 1855 -->
                <xs:enumeration value="COA"></xs:enumeration>
            </xs:restriction>
</xs:simpleType>
<xs:simpleType name="defuzzifierType">
       <xs:union memberTypes="standardDefuzzifierType customLabelType"/>
</xs:simpleType>
<xs:simpleType name="customLabelType">
            <xs:restriction base="xs:string">
                <xs:pattern value="custom_\S*"/>
            </xs:restriction>
</xs:simpleType>
<xs:simpleType name="standardAccumulationType">
          <xs:restriction base="xs:string">
              <!-- MAX reference: Equation (A.35) from IEEE Std 1855 -->
              <xs:enumeration value="MAX"></xs:enumeration>
              <!-- PROBOR reference: Equation (A.36) from IEEE Std 1855 -->
              <xs:enumeration value="PROBOR"></xs:enumeration>
              <!-- BSUM reference: Equation (A.37) from IEEE Std 1855 -->
              <xs:enumeration value="BSUM"></xs:enumeration>
              <!-- DRS reference: Equation (A.38) from IEEE Std 1855 -->
              <xs:enumeration value="DRS"></xs:enumeration>
              <!-- ESUM reference: Equation (A.39) from IEEE Std 1855 -->
              <xs:enumeration value="ESUM"></xs:enumeration>
```

```
                <!-- HSUM reference: Equation (A.40) from IEEE Std 1855 -->
                <xs:enumeration value="HSUM"></xs:enumeration>
                <!-- NILMAX reference: Equation (A.41) from IEEE Std 1855 -->
                <xs:enumeration value="NILMAX"></xs:enumeration>
            </xs:restriction>
</xs:simpleType>
<xs:simpleType name="accumulationType">
            <xs:union memberTypes="standardAccumulationType customLabelType"/>
</xs:simpleType>
<xs:simpleType name="typeType">
            <xs:restriction base="xs:string">
                <xs:pattern value="input|output|INPUT|OUTPUT|Input|Output"/>
            </xs:restriction>
</xs:simpleType>
<xs:simpleType name="tskOrderType">
            <xs:restriction base="xs:int">
                <xs:pattern value="0|1"/>
            </xs:restriction>
</xs:simpleType>
<xs:simpleType name="standardActivationMethodType">
            <xs:restriction base="xs:string">
                <!-- MIN reference: Equation (A.28) from IEEE Std 1855 -->
                <xs:enumeration value="MIN"></xs:enumeration>
                <!-- PROD reference: Equation (A.29) from IEEE Std 1855 -->
                <xs:enumeration value="PROD"></xs:enumeration>
                <!-- BDIF reference: Equation (A.30) from IEEE Std 1855 -->
                <xs:enumeration value="BSUM"></xs:enumeration>
                <!-- DRP reference: Equation (A.31) from IEEE Std 1855 -->
                <xs:enumeration value="DRS"></xs:enumeration>
                <!-- EPROD reference: Equation (A.32) from IEEE Std 1855 -->
                <xs:enumeration value="EPROD"></xs:enumeration>
                <!-- HPROD reference: Equation (A.33) from IEEE Std 1855 -->
                <xs:enumeration value="HPROD"></xs:enumeration>
                <!-- NILMIN reference: Equation (A.34) from IEEE Std 1855 -->
                <xs:enumeration value="NILMIN"></xs:enumeration>
            </xs:restriction>
</xs:simpleType>
<xs:simpleType name="combinationType">
            <xs:union memberTypes="standardCombinationType customLabelType"/>
</xs:simpleType>
 <xs:simpleType name="standardCombinationType">
            <xs:restriction base="xs:string">
                <!-- WA reference: Equation (A.47) from IEEE Std 1855 -->
                <xs:enumeration value="WA"></xs:enumeration>
            </xs:restriction>
</xs:simpleType>
<xs:simpleType name="activationMethodType">
            <xs:union memberTypes="standardActivationMethodType customLabelType"/>
</xs:simpleType>
<xs:simpleType name="standardTnormType">
            <xs:restriction base="xs:string">
                <!-- MIN reference: Equation (A.14) from IEEE Std 1855 -->
                <xs:enumeration value="MIN"></xs:enumeration>
                <!-- PROD reference: Equation (A.15) from IEEE Std 1855 -->
                <xs:enumeration value="PROD"></xs:enumeration>
                <!-- BDIF reference: Equation (A.16) from IEEE Std 1855 -->
                <xs:enumeration value="BSUM"></xs:enumeration>
                <!-- DRP reference: Equation (A.17) from IEEE Std 1855 -->
                <xs:enumeration value="DRS"></xs:enumeration>
                <!-- EPROD reference: Equation (A.18) from IEEE Std 1855 -->
                <xs:enumeration value="EPROD"></xs:enumeration>
                <!-- HPROD reference: Equation (A.19) from IEEE Std 1855 -->
                <xs:enumeration value="HPROD"></xs:enumeration>
                <!-- NILMIN reference: Equation (A.20) from IEEE Std 1855 -->
                <xs:enumeration value="NILMIN"></xs:enumeration>
            </xs:restriction>
</xs:simpleType>
<xs:simpleType name="andMethodType">
            <xs:union memberTypes="standardTnormType customLabelType"/>
</xs:simpleType>
<xs:simpleType name="standardTconormType">
```

```
            <xs:restriction base="xs:string">
               <!-- MAX reference: Equation (A.21) from IEEE Std 1855 -->
               <xs:enumeration value="MAX"></xs:enumeration>
               <!-- PROBOR reference: Equation (A.22) from IEEE Std 1855 -->
               <xs:enumeration value="PROBOR"></xs:enumeration>
               <!-- BSUM reference: Equation (A.23) from IEEE Std 1855 -->
               <xs:enumeration value="BSUM"></xs:enumeration>
               <!-- DRS reference: Equation (A.24) from IEEE Std 1855 -->
               <xs:enumeration value="DRS"></xs:enumeration>
               <!-- ESUM reference: Equation (A.25) from IEEE Std 1855 -->
               <xs:enumeration value="ESUM"></xs:enumeration>
               <!-- HSUM reference: Equation (A.26) from IEEE Std 1855 -->
               <xs:enumeration value="HSUM"></xs:enumeration>
               <!-- NILMAX reference: Equation (A.27) from IEEE Std 1855 -->
               <xs:enumeration value="NILMAX"></xs:enumeration>
            </xs:restriction>
</xs:simpleType>
<xs:simpleType name="orMethodType">
            <xs:union memberTypes="standardTconormType customLabelType"/>
</xs:simpleType>
<xs:simpleType name="connectorType">
            <xs:restriction base="xs:string">
                <xs:pattern value="and|AND|OR|or"/>
            </xs:restriction>
 </xs:simpleType>
  <xs:simpleType name="standardModifierType">
            <xs:restriction base="xs:string">
               <!-- above reference: Equation (A.1) from IEEE Std 1855 -->
               <xs:enumeration value="above"></xs:enumeration>
               <!-- any reference: Equation (A.2) from IEEE Std 1855 -->
               <xs:enumeration value="any"></xs:enumeration>
               <!-- below reference: Equation (A.3) from IEEE Std 1855 -->
               <xs:enumeration value="below"></xs:enumeration>
               <!-- extremely reference: Equation (A.4) from IEEE Std 1855 -->
               <xs:enumeration value="extremely"></xs:enumeration>
               <!-- intensify reference: Equation (A.5) from IEEE Std 1855 -->
               <xs:enumeration value="intensify"></xs:enumeration>
               <!-- more_or_less reference: Equation (A.6) from IEEE Std 1855 -->
               <xs:enumeration value="more_or_less"></xs:enumeration>
               <!-- norm reference: Equation (A.7) from IEEE Std 1855 -->
               <xs:enumeration value="norm"></xs:enumeration>
               <!-- not reference: Equation (A.8) from IEEE Std 1855 -->
               <xs:enumeration value="not"></xs:enumeration>
               <!-- plus reference: Equation (A.9) from IEEE Std 1855 -->
               <xs:enumeration value="plus"></xs:enumeration>
               <!-- seldom reference: Equation (A.10) from IEEE Std 1855 -->
               <xs:enumeration value="seldom"></xs:enumeration>
               <!-- slightly reference: Equation (A.11) from IEEE Std 1855 -->
               <xs:enumeration value="slightly"></xs:enumeration>
               <!-- somewhat reference: Equation (A.12) from IEEE Std 1855 -->
               <xs:enumeration value="somewhat"></xs:enumeration>
               <!-- very reference: Equation (A.13) from IEEE Std 1855 -->
               <xs:enumeration value="very"></xs:enumeration>
             </xs:restriction>
 </xs:simpleType>
 <xs:simpleType name="modifierType">
            <xs:union memberTypes="standardModifierType customLabelType"/>
</xs:simpleType>
<xs:simpleType name="weightType">
             <xs:restriction base="xs:float">
               <xs:minInclusive value="0"/>
               <xs:maxInclusive value="1.0"/>
             </xs:restriction>
</xs:simpleType>
<xs:simpleType name="interpolationMethodType">
            <xs:restriction base="xs:string">
               <!-- Linear reference: [B11] from IEEE Std 1855 -->
               <xs:enumeration value="linear"></xs:enumeration>
               <!-- Lagrange reference: [B10] from IEEE Std 1855 -->
               <xs:enumeration value="lagrange"></xs:enumeration>
               <!-- Spline reference: [B12] from IEEE Std 1855 -->
```

```
                <xs:enumeration value="spline"></xs:enumeration>
             </xs:restriction>
   </xs:simpleType>
   <xs:simpleType name="monotonicInterpolationMethodType">
             <xs:restriction base="xs:string">
                <!-- Linear reference: [B11] from IEEE Std 1855 -->
                <xs:enumeration value="linear"></xs:enumeration>
                <!-- Cubic reference: [B9] from IEEE Std 1855 -->
                <xs:enumeration value="cubic"></xs:enumeration>
             </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="networkAddressType">
            <xs:union memberTypes="InetAddressIPv4 InetAddressIPv6 macAddressType
                     hostnameType"/>
   </xs:simpleType>

   <xs:simpleType name="InetAddressIPv4">
 <xs:annotation>
  <xs:documentation>
   InetAddressIPv4 as defined in [B18] at pages 7 - 8 from IEEE Std 1855.
  </xs:documentation>
 </xs:annotation>
 <xs:restriction base="xs:string">
  <xs:pattern value="((1?[0-9]?[0-9]|2[0-4][0-9]|25[0-5]).){3}(1?[0-9]?[0-9]|2[0-4][0-
                     9]|25[0-5])"/>
  <xs:minLength value="7"/>
  <xs:maxLength value="15"/>
 </xs:restriction>
</xs:simpleType>

<xs:simpleType name="InetAddressIPv6">
 <xs:annotation>
  <xs:documentation>
   InetAddressIPv6 as defined in [B18] at page 8 from IEEE Std 1855.
  </xs:documentation>
 </xs:annotation>
 <xs:restriction base="xs:string">
  <xs:pattern value="(([0-9a-fA-F]{1,4}:){6})(([0-9a-fA-F]{1,4}:[0-9a-fA-F]{1,4})|([0-
                     9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}))"/>
 </xs:restriction>
</xs:simpleType>

 <xs:simpleType name="macAddressType">
       <xs:restriction base="xs:string">
           <xs:pattern value="([0-9a-fA-F]{2}:){5}[0-9a-fA-F]{2}" />
           <xs:whiteSpace value="collapse" />
       </xs:restriction>
   </xs:simpleType>

   <xs:simpleType name="hostnameType">
       <xs:restriction base="xs:string">
           <xs:maxLength value="255" />
           <xs:pattern
               value="[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\.[A-Za-z]([A-Za-z0-9-]*[A-Za-
                      z0-9])?)*" />
           <xs:whiteSpace value="collapse" />
       </xs:restriction>
   </xs:simpleType>

</xs:schema>
```

Annex C

(normative)

Examples

C.1 Mamdani and TSK versions

The Tipper System is used to regulate the tipping in, for example, a restaurant [B1]. It has two variables for input (food and service) and one variable for output (tip). In this subclause, we consider a Mamdani and a Takagi-Sugeno-Kang (TSK) version of the Tipper System.

The following example shows the FML code for the Mamdani version of the Tipper System as described in [B1].

```
<?xml version="1.0" encoding="UTF-8"?>
<fuzzySystem name="newSystem" networkAddress="127.0.0.1">
    <knowledgeBase>
        <fuzzyVariable name="food" domainleft="0.0" domainright="10.0" scale=""
type="input">
            <fuzzyTerm name="delicious" complement="false">
                <leftLinearShape param1="5.5" param2="10.0"/>
            </fuzzyTerm>
            <fuzzyTerm name="rancid" complement="false">
                <triangularShape param1="0.0" param2="2" param3="5.5"/>
            </fuzzyTerm>
        </fuzzyVariable>
        <fuzzyVariable name="service" domainleft="0.0" domainright="10.0" scale=""
type="input">
            <fuzzyTerm name="excellent" complement="false">
                <leftGaussianShape param1="10.0" param2="2.0"/>
            </fuzzyTerm>
            <fuzzyTerm name="good" complement="false">
                <piShape param1="5.0" param2="4.0"/>
            </fuzzyTerm>
            <fuzzyTerm name="poor" complement="false">
                <rightGaussianShape param1="0.0" param2="2.0"/>
            </fuzzyTerm>
        </fuzzyVariable>
        <fuzzyVariable name="tip" domainleft="0.0" domainright="20.0" scale="null"
defaultValue="0.0" accumulation="MAX" defuzzifier="COG" type="output">
            <fuzzyTerm name="average" complement="false">
                <triangularShape param1="5.0" param2="10.0" param3="15.0"/>
            </fuzzyTerm>
            <fuzzyTerm name="cheap" complement="false">
                <triangularShape param1="0.0"param2="5.0" param3="10.0"/>
            </fuzzyTerm>
            <fuzzyTerm name="generous" complement="false">
                <triangularShape param1="10.0" param2="15.0" param3="20.0"/>
            </fuzzyTerm>
        </fuzzyVariable>
    </knowledgeBase>
    <mamdaniRuleBase name="rulebase1" andMethod="MIN" orMethod="MAX"
activationMethod="MIN">
        <rule name="reg1" connector="or" orMethod="MAX" weight="1.0">
            <antecedent>
                <clause>
                    <variable>food</variable>
                    <term>rancid</term>
                </clause>
                <clause modifier="very">
                    <variable>service</variable>
                    <term>poor</term>
```

```
                </clause>
            </antecedent>
            <consequent>
                <then>
                  <clause>
                      <variable>tip</variable>
                      <term>cheap</term>
                  </clause>
                </then>
            </consequent>
        </rule>
        <rule name="reg2" connector="or" orMethod="MAX" weight="1.0">
            <antecedent>
                <clause>
                    <variable>service</variable>
                    <term>good</term>
                </clause>
            </antecedent>
            <consequent>
                <then>
                  <clause>
                      <variable>tip</variable>
                      <term>average</term>
                  </clause>
                </then>
            </consequent>
        </rule>
        <rule name="reg3" connector="or" orMethod="MAX" weight="1.0">
            <antecedent>
                <clause>
                    <variable>service</variable>
                    <term>excellent</term>
                </clause>
                <clause>
                    <variable>food</variable>
                    <term>delicious</term>
                </clause>
            </antecedent>
            <consequent>
                <then>
                  <clause>
                      <variable>tip</variable>
                      <term>generous</term>
                  </clause>
                </then>
            </consequent>
        </rule>
    </mamdaniRuleBase>
</fuzzySystem>
```

The following listing shows the FML code for a TSK version of the Tipper System.

```
<?xml version="1.0" encoding="UTF-8"?>
<fuzzySystem name="tipper" networkAddress="127.0.0.1">
    <knowledgeBase>
        <fuzzyVariable name="food" domainleft="0.0" domainright="10.0" scale=""
type="input">
            <fuzzyTerm name="delicious" complement="false">
                <leftLinearShape param1="5.5" param2="10.0"/>
            </fuzzyTerm>
            <fuzzyTerm name="rancid" complement="false">
                <triangularShape param1="0.0" param2="2" param3="5.5"/>
            </fuzzyTerm>
        </fuzzyVariable>
        <fuzzyVariable name="service" domainleft="0.0" domainright="10.0" scale=""
type="input">
            <fuzzyTerm name="excellent" complement="false">
                <leftGaussianShape param1="10.0" param2="2.0"/>
            </fuzzyTerm>
            <fuzzyTerm name="good" complement="false">
```

```
            <piShape param1="5.0" param2="4.0"/>
        </fuzzyTerm>
        <fuzzyTerm name="poor" complement="false">
            <rightGaussianShape param1="0.0" param2="2.0"/>
        </fuzzyTerm>
    </fuzzyVariable>
    <tskVariable name="tip" scale="null" combination="WA" type="output">
        <tskTerm name="average" order="0">
            <tskValue>1.6</tskValue>
        </tskTerm>
        <tskTerm name="cheap" order="1">
            <tskValue>1.9</tskValue>
            <tskValue>5.6</tskValue>
            <tskValue>6.0</tskValue>
        </tskTerm>
        <tskTerm name="generous" order="1">
            <tskValue>0.6</tskValue>
            <tskValue>1.3</tskValue>
            <tskValue>1.0</tskValue>
        </tskTerm>
    </tskVariable>
</knowledgeBase>
<tskRuleBase name="No1" andMethod="MIN" orMethod="MAX" activationMethod="MIN">
    <tskRule name="reg1" connector="or" orMethod="MAX" weight="1.0">
        <antecedent>
            <clause>
                <variable>food</variable>
                <term>rancid</term>
            </clause>
            <clause>
                <variable>service</variable>
                <term>poor</term>
            </clause>
        </antecedent>
        <tskConsequent>
            <tskThen>
              <tskClause>
                <variable>tip</variable>
                <term>cheap</term>
              </tskClause>
            </tskThen>
        </tskConsequent>
    </tskRule>
    <tskRule name="reg2" connector="or" orMethod="MAX" weight="1.0">
        <antecedent>
            <clause>
                <variable>service</variable>
                <term>good</term>
            </clause>
        </antecedent>
        <tskConsequent>
            <tskThen>
              <tskClause>
                <variable>tip</variable>
                <term>average</term>
              </tskClause>
            </tskThen>
        </tskConsequent>
    </tskRule>
    <tskRule name="reg3" connector="or" orMethod="MAX" weight="1.0">
        <antecedent>
            <clause>
                <variable>food</variable>
                <term>delicious</term>
            </clause>
            <clause>
                <variable>service</variable>
                <term>excellent</term>
            </clause>
        </antecedent>
        <tskConsequent>
```

```
                <tskThen>
                  <tskClause>
                      <variable>tip</variable>
                      <term>generous</term>
                  </tskClause>
              </tskThen>
            </tskConsequent>
          </tskRule>
    </tskRuleBase>
</fuzzySystem>
```

C.2 Japanese diet assessment system

This system is aimed at evaluating a dietary healthy level of Japanese food intake for each day. A balanced diet must contain correctly proportioned carbohydrates, protein, fat, vitamins, mineral salts, and fiber because each nutrient plays a relevant role in supporting humans' everyday physical activities. Therefore, this system uses the following input variables: PCC to describe the linguistic meanings of Percentage of Calories from Carbohydrate, PCP for the Percentage of Calories from Protein, PCF for the Percentage of Calories from Fat, PCR for the Percentage of Caloric Ratio (PCR), and FGB for the Food Group Balance. Based on the information from PCC, PCP, PCF, PCR, and FGB, the Dietary Healthy Level (DHL) is obtained.

Below, a portion of the FML code used to model the diet evaluation system presented in [B15].

```
<?xml version="1.0"?>
<fuzzySystem networkAddress="127.0.0.1" name="">
  <knowledgeBase>
    <fuzzyVariable domainleft="0" domainright="100" name="PCC" scale="percentage"
type="input">
      <fuzzyTerm name="Low" complement="false">
        <trapezoidShape Param1="0" Param2="0" Param3="55" Param4="60" />
      </fuzzyTerm>
      <fuzzyTerm name="Medium" complement="false">
        <trapezoidShape Param1="55" Param2="60" Param3="65" Param4="70" />
      </fuzzyTerm>
      <fuzzyTerm name="High" complement="false">
        <trapezoidShape Param1="65" Param2="70" Param3="100" Param4="100" />
      </fuzzyTerm>
    </fuzzyVariable>
    <fuzzyVariable domainleft="0" domainright="100" name="PCP" scale="percentage"
type="input">
      <fuzzyTerm name="Low" complement="false">
        <trapezoidShape Param1="0" Param2="0" Param3="10" Param4="15" />
      </fuzzyTerm>
      <fuzzyTerm name="Medium" complement="false">
        <trapezoidShape Param1="10" Param2="15" Param3="18" Param4="21" />
      </fuzzyTerm>
      <fuzzyTerm name="High" complement="false">
        <trapezoidShape Param1="18" Param2="21" Param3="100" Param4="100" />
      </fuzzyTerm>
    </fuzzyVariable>
    <fuzzyVariable domainleft="0" domainright="100" name="PCF" scale="percentage"
type="input">
      <fuzzyTerm name="Low" complement="false">
        <trapezoidShape Param1="0" Param2="0" Param3="15" Param4="20" />
      </fuzzyTerm>
      <fuzzyTerm name="Medium" complement="false">
        <trapezoidShape Param1="15" Param2="20" Param3="24" Param4="30" />
      </fuzzyTerm>
      <fuzzyTerm name="High" complement="false">
        <trapezoidShape Param1="24" Param2="30" Param3="100" Param4="100" />
      </fuzzyTerm>
    </fuzzyVariable>
    <fuzzyVariable domainleft="0" domainright="7" name="FGB" scale="percentage"
type="input">
      <fuzzyTerm name="Low" complement="false">
        <trapezoidShape Param1="0" Param2="0" Param3="1" Param4="3" />
```

```
      </fuzzyTerm>
      <fuzzyTerm name="Medium" complement="false">
        <trapezoidShape Param1="1" Param2="3" Param3="4" Param4="6" />
      </fuzzyTerm>
      <fuzzyTerm name="High" complement="false">
        <trapezoidShape Param1="4" Param2="6" Param3="7" Param4="7" />
      </FuzzyTerm>
    </fuzzyVariable>
    <fuzzyVariable domainleft="0" domainright="200" name="PCR" scale="percentage"
type="input">
      <fuzzyTerm name="Low" complement="false">
        <trapezoidShape Param1="0" Param2="0" Param3="85" Param4="95" />
      </fuzzyTerm>
      <fuzzyTerm name="Medium" complement="false">
        <trapezoidShape Param1="85" Param2="95" Param3="105" Param4="115" />
      </fuzzyTerm>
      <fuzzyTerm name="High" complement="false">
        <trapezoidShape Param1="105" Param2="115" Param3="200" Param4="200" />
      </fuzzyTerm>
    </fuzzyVariable>
    <fuzzyVariable domainleft="0" domainright="10" name="DHL" scale="degree"
type="output">
      <fuzzyTerm name="VeryLow" complement="false">
        <trapezoidShape Param1="0" Param2="0" Param3="1.5" Param4="2.5" />
      </fuzzyTerm>
      <fuzzyTerm name="Low" complement="false">
        <trapezoidShape Param1="1.5" Param2="2.5" Param3="3.5" Param4="4.5" />
      </fuzzyTerm>
      <fuzzyTerm name="Medium" complement="false">
        <trapezoidShape Param1="3.5" Param2="4.5" Param3="5.5" Param4="6.5" />
      </fuzzyTerm>
      <fuzzyTerm name="High" complement="false">
        <trapezoidShape Param1="5.5" Param2="6.5" Param3="7.5" Param4="8.5" />
      </fuzzyTerm>
      <fuzzyTerm name="VeryHigh" complement="false">
        <trapezoidShape Param1="7.5" Param2="8.5" Param3="10" Param4="10" />
      </fuzzyTerm>
    </fuzzyVariable>
  </knowledgeBase>
  <mamdaniRuleBase activationMethod="MIN" andMethod="MIN" orMethod="MAX"
name="RuleBase1">
    <rule name="Rule1" connector="and" weight="1" andMethod="MIN">
      <antecedent>
        <clause>
          <variable>PCC</variable>
          <term>Low</term>
        </clause>
        <clause>
          <variable>PCP</variable>
          <term>Low</term>
        </clause>
        <clause>
          <variable>PCF</variable>
          <term>Low</term>
        </clause>
        <clause>
          <variable>FGB</variable>
          <term>Low</term>
        </clause>
        <clause>
          <variable>PCR</variable>
          <term>Low</term>
        </clause>
      </antecedent>
      <consequent>
        <then>
          <clause>
            <variable>DHL</variable>
            <term>VeryLow</term>
          </clause>
        </then>
```

```
      </consequent>
    </rule>
    ..........................................................................
    <rule name="Rule243" connector="and" weight="1" andMethod="MIN">
      <antecedent>
        <clause>
          <variable>PCC</variable>
          <term>High</term>
        </clause>
        <clause>
          <variable>PCP</variable>
          <term>High</term>
        </clause>
        <clause>
          <variable>PCF</variable>
          <term>High</term>
        </clause>
        <clause>
          <variable>FGB</variable>
          <term>High</term>
        </clause>
        <clause>
          <variable>PCR</variable>
          <term>High</term>
        </clause>
      </antecedent>
      <consequent>
        <then>
          <clause>
            <variable>DHL</variable>
            <term>Low</term>
          </clause>
        </then>
      </consequent>
    </rule>
  </mamdaniRuleBase>
</fuzzySystem>
```

Annex D

(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] Acampora, G., "Fuzzy Markup Language: A XML Based Language for Enabling Full Interoperability in Fuzzy Systems Design." In *On the Power of Fuzzy Markup Language*, edited by G. Acampora, V. Loia, C.-S. Lee, and M.-H. Wang. Berlin, Germany: Springer Berlin Heidelberg, 2013, pp. 17–31.

[B2] Acampora, G., V. Loia, and A. Vitiello. "An Enhanced Visual Environment for Designing, Testing and Developing FML-Based Fuzzy Systems." In *On the Power of Fuzzy Markup Language*, edited by G. Acampora, V. Loia, C.-S. Lee, and M.-H. Wang. Berlin, Germany: Springer Berlin Heidelberg, 2013, pp. 51–70.

[B3] Acampora, G., V. Loia, and A. Vitiello. "Distributing Fuzzy Reasoning through Fuzzy Markup Language: An Application to Ambient Intelligence." In *On the Power of Fuzzy Markup Language*, edited by G. Acampora, V. Loia, C.-S. Lee, and M.-H. Wang. Berlin, Germany: Springer Berlin Heidelberg, 2013, pp. 33–50.

[B4] Acampora, G., V. Loia, and A. Vitiello, "Improving game bot behaviours through timed emotional intelligence." *Knowledge-Based System* 34, (2012): 97–113.

[B5] Angelov, P., and R. Yager. "Simplified fuzzy rule-based systems using non-parametric antecedents and relative data density," *2011 IEEE Workshop on Evolving and Adaptive Intelligent Systems* (EAIS 2011), pp. 62–69, Apr. 11–15, 2011.

[B6] Bray, T., D. Hollander, A. Layman, R. Tobin, and H. S. Thompson, "Namespaces in XML 1.0." W3C, Dec. 2009.

[B7] Di Stefano, B. N., "On the Need of a Standard Language for Designing Fuzzy Systems." In *On the Power of Fuzzy Markup Language*, edited by G. Acampora, V. Loia, C.-S. Lee, and M.-H. Wang. Berlin, Germany: Springer Berlin Heidelberg, 2013, pp. 3–15.

[B8] Eclipse documentation, Icons used in the XML Schema editor. Available: http://help.eclipse.org/luna/topic/org.eclipse.wst.xsdeditor.doc.user/topics/rxsdicons.html?cp=65_2_3_6.

[B9] Fritsch, F.. N., and R. E. Carlson, "Monotone Piecewise Cubic Interpolation," *SIAM Journal on Numerical Analysis* 17, no.2 (1980): 238–246.

[B10] Hazewinkel, M., "Lagrange interpolation formula." *Encyclopedia of Mathematics,* Springer, ISBN 978-1-55608-010-4, ed. (2001).

[B11] Hazewinkel, M., "Linear interpolation." *Encyclopedia of Mathematics*, Springer, ISBN 978-1-55608-010-4, ed. (2001).

[B12] Hazewinkel, M., "Spline interpolation." *Encyclopedia of Mathematics*, Springer, ISBN 978-1-55608-010-4, ed. (2001).

[B13] Huang, H.-D., G. Acampora, V. Loia, C.-S. Lee, H. Hagras, M.-H. Wang, H.-Y. Kao, and J.-G. Chang, "Fuzzy Markup Language for Malware Behavioral Analysis." In *On the Power of Fuzzy Markup Language*, edited by G. Acampora, V. Loia, C.-S. Lee, and M.-H. Wang. Berlin, Germany: Springer Berlin Heidelberg, 2013, pp. 113–132.

[B14] IEC 61131-3:1993, Programmable controllers – Part 7: Fuzzy control programming.

[B15] Kurozumi, K., S.-T. Lan, M.-H. Wang, C.-S. Lee, M. Kawaguchi, S. Tsumoto, and H. Tsuji, "FML-based Japanese diet assessment system," *2013 IEEE International Conference on Fuzzy Systems* (FUZZ-IEEE 2013), pp. 1–6, July 7–10, 2013.

[B16] Mamdani, E. H., and S. Assilian. "An experiment in linguistic synthesis with a fuzzy logic controller." *International journal of man-machine studies* 7, no.1 (1975): 1–13.

[B17] RFC 2119, Key words for use in RFCs to Indicate Requirement Levels. Available: http://www.ietf.org/rfc/rfc2119.txt.

[B18] RFC 4001, Textual Conventions for Internet Network Addresses. Available: https://www.ietf.org/rfc/rfc4001.txt.

[B19] Shinghal, R., *Introduction to Fuzzy Logic*. PHI Learning Pvt. Ltd., 2012.

[B20] Takagi, T., and M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 1, pp. 116–132, 1985.

[B21] Tsukamoto, Y., "An approach to fuzzy reasoning method," *Advances in fuzzy set theory and applications*, (1979): 137–149.

[B22] Wang, M.-H., C.-S. Lee, K.-L. Hsieh, C.-Y. Hsu, G. Acampora, C.-C. Chang, "Ontology-based multi-agents for intelligent healthcare applications," *Journal of Ambient Intelligence and Humanized Computing*, vol. 1, no.2, pp. 111–131, 2010.

[B23] XML Schema Part 0: Primer. Available: http://www.w3.org/TR/xmlschema-0/.

[B24] XML Schema Tutorial. Available: http://www.xfront.com.

[B25] XML Schema Tutorial, Part 1. Available: www.liquid-technologies.com/Tutorials/XmlSchemas/XsdTutorial_01.aspx.

[B26] Zadeh, L. A., "Fuzzy logic and its application to approximate reasoning," *Information Processing*, vol. 74, no.3, pp. 591–594, 1974.

[B27] Zadeh, L. A., "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353, 1965.

[B28] Zadeh, L. A., "Making computers think like people," *IEEE Spectrum*, vol. 8, pp. 26–32, 1984.

Consensus

WE BUILD IT.

Connect with us on:

Facebook: https://www.facebook.com/ieeesa

Twitter: @ieeesa

LinkedIn: http://www.linkedin.com/groups/IEEESA-Official-IEEE-Standards-Association-1791118

IEEE-SA Standards Insight blog: http://standardsinsight.com

YouTube: IEEE-SA Channel

IEEE
standards.ieee.org
Phone: +1 732 981 0060 Fax: +1 732 562 1571