

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

Copyright © 2016 by The Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 27 May 2016. Printed in the United States of America.

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

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

IEEE prohibits discrimination, harassment, and bullying.

For more information, visit <http://www.ieee.org/web/aboutus/whatis/policies/p9-26.html>.

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

Important Notices and Disclaimers Concerning IEEE Standards Documents

IEEE 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

1. Overview	9
2. Normative references.....	12
3. Definitions, acronyms, and abbreviations	12
4. A fuzzy system model for enabling a standard FML-based representation.....	14
5. FML Schema definition.....	21
6. FML Synthesis: XSLT Documents and DOM API	53
7. Compatibility with IEC 61131-7	54
8. Conformance	56
9. Extensibility.....	57
Annex A (normative) Fuzzy logic theory	59
Annex B (normative) XML Schema.....	67
Annex C (normative) Examples	80
C.1 Mamdani and TSK versions.....	80
C.2 Japanese diet assessment system.....	83
Annex D (informative) Bibliography	86

IEEE Standard for Fuzzy Markup Language

IMPORTANT NOTICE: IEEE Standards documents are not intended to ensure safety, security, health, or environmental protection, or ensure against interference with or from other devices or networks. Implementers of IEEE Standards documents are responsible for determining and complying with all appropriate safety, security, environmental, health, and interference protection practices and all applicable laws and regulations.

This IEEE document is made available for use subject to important notices and legal disclaimers. These notices and disclaimers appear in all publications containing this document and may be found under the heading “Important Notice” or “Important Notices and Disclaimers Concerning IEEE Documents.” They can also be obtained on request from IEEE or viewed at <http://standards.ieee.org/IPR/disclaimers.html>.

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:

- **Blue** shows XML elements.
- **Red** shows XML attributes.

In example code, the used convention is as follows:

- **Blue** shows XML tags.
- **Red** shows XML attributes.
- **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 $l: V \rightarrow \Sigma_V$, E be a set of unordered distinct vertices called edges, and a be a total function $a: E \rightarrow \Sigma_E$. 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*.

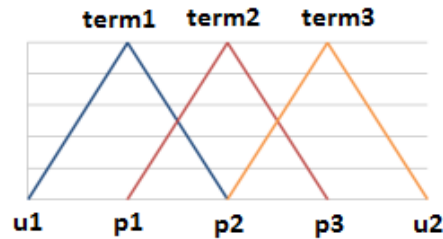


Figure 1—Fuzzy variable *var* [B2]

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
$n_{var_1} = \text{FuzzyVariable}$	$a_{var_1} = a_{var_5} = a_{var_{10}} =$ $a_{var_{15}} = \text{name}$	$t_{var_1} = \text{var}$
$n_{var_2} = n_{var_3} = n_{var_4} = \text{FuzzyTerm}$	$a_{var_2} = \text{domainLeft}$	$t_{var_2} = u1$
$n_{var_5} = n_{var_6} = n_{var_7} = \text{TriangularShape}$	$a_{var_3} = \text{domainRight}$	$t_{var_3} = u2$
	$a_{var_4} = \text{scale}$	$t_{var_4} = \text{unit}$
	$a_{var_6} = a_{var_{11}} = a_{var_{16}} =$ not	$t_{var_5} = \text{term1}, t_{var_{10}} = \text{term2},$ $t_{var_{15}} = \text{term3}$
	$a_{var_7} = a_{var_{12}} = a_{var_{17}} =$ param1	$t_{var_6} = t_{var_{11}} = t_{var_{16}} = \text{false}$
	$a_{var_8} = a_{var_{13}} = a_{var_{18}} =$ 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 :

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

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

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

$$(n_{var_5}, a_{var_7}), \dots, (a_{var_1}, t_{var_1}), \dots, (a_{var_9}, t_{var_9})\}$$

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:

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

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

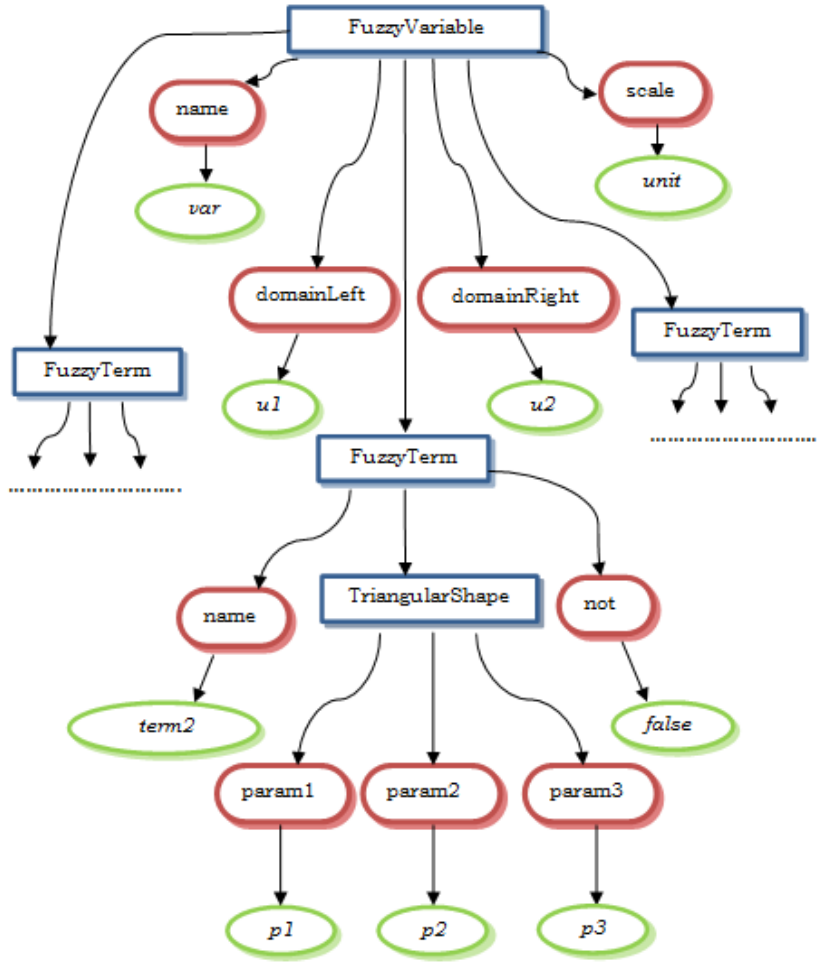


Figure 2—Labeled tree for the fuzzy variable *var*

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

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:

$$E_{fr} = \{(n_{fr_1}, n_{fr_2}), (n_{fr_1}, n_{fr_9}), (n_{fr_1}, a_{fr_1}), \dots, (n_{fr_1}, a_{fr_{15}}), (a_{fr_1}, t_{fr_1}), \dots, (a_{fr_5}, t_{fr_5}), (n_{fr_2}, n_{fr_3}), (n_{fr_2}, n_{fr_6}), \dots, (n_{fr_4}, t_{fr_6}), \dots, (n_{fr_5}, n_{fr_7}), \dots\}$$

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

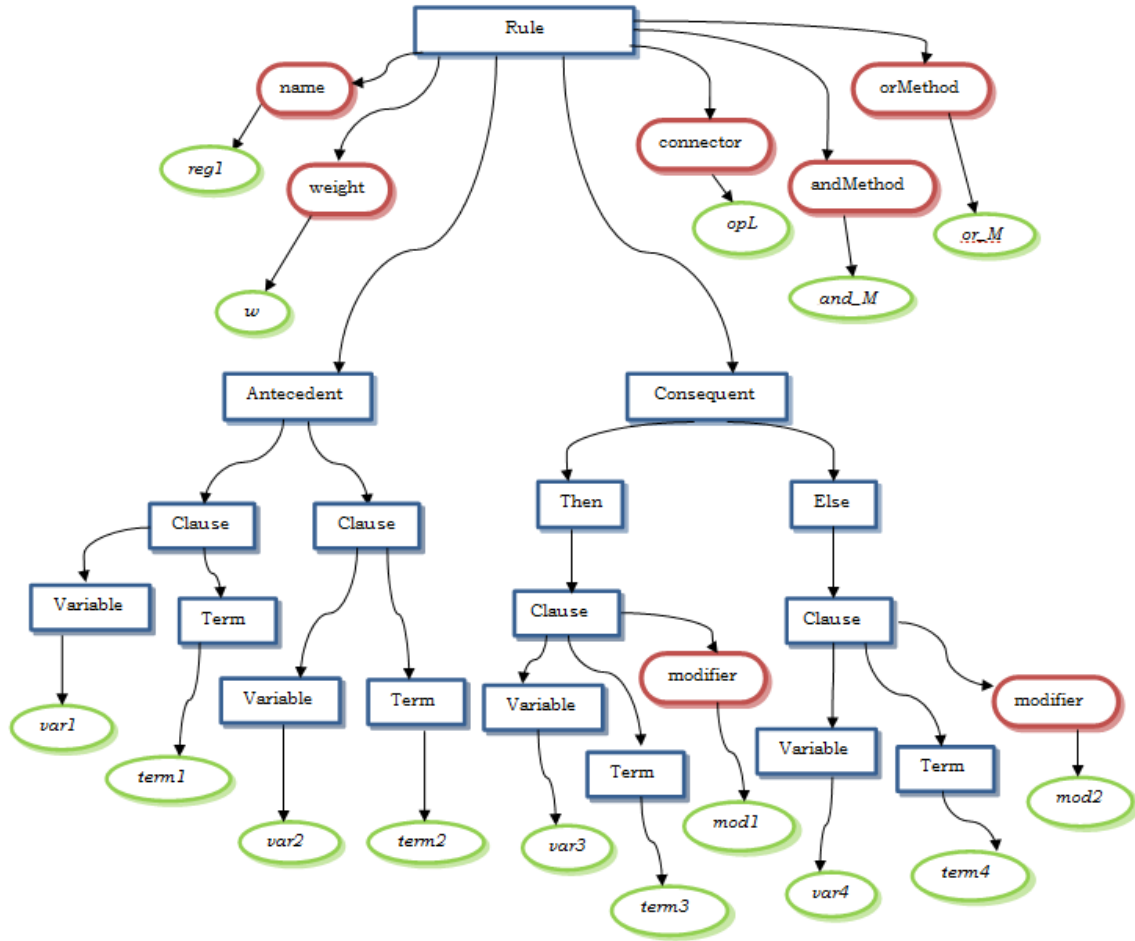


Figure 3—Labeled tree for the rule reg1

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:

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

$$E_{RB} = \bigcup_i E_{R_i} \cup \bigcup_i \{(n_{RB}, n_{R_i})\} \cup E_{RA}$$

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.

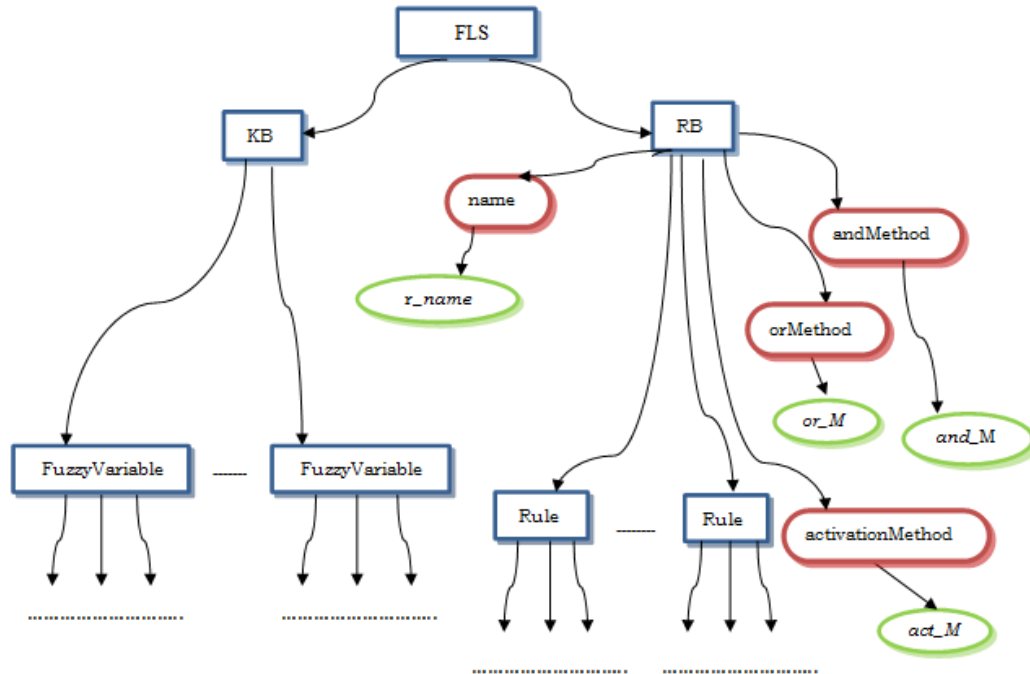


Figure 4—A partial FLS labeled tree

4.2 Benefits of FLS labeled trees

FML achieves its purpose by using the labeled tree approach 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**, **tsukamotoRuleBase**, **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: — $(([0-1]?[0-9]?[0-9] 2?[0-5]?[0-5])\backslash.\{3\}([0-1]?[0-9]?[0-9] 2?[0-5]?[0-5])$ for an address IPv4 as defined in [B18] at pages 7–8; — $(([0-9a-fA-F]\{1,4\}:)\{6\})(([0-9a-fA-F]\{1,4\}:[0-9a-fA-F]\{1,4\}) ([0-9]\{1,3\}\backslash.[0-9]\{1,3\}\backslash.[0-9]\{1,3\}\backslash.[0-9]\{1,3\}))$ 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; — $[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?(\backslash.[A-Za-z]([A-Za-z0-9-]*[A-Za-z0-9])?)^*$ for a host name.	127.0.0.1

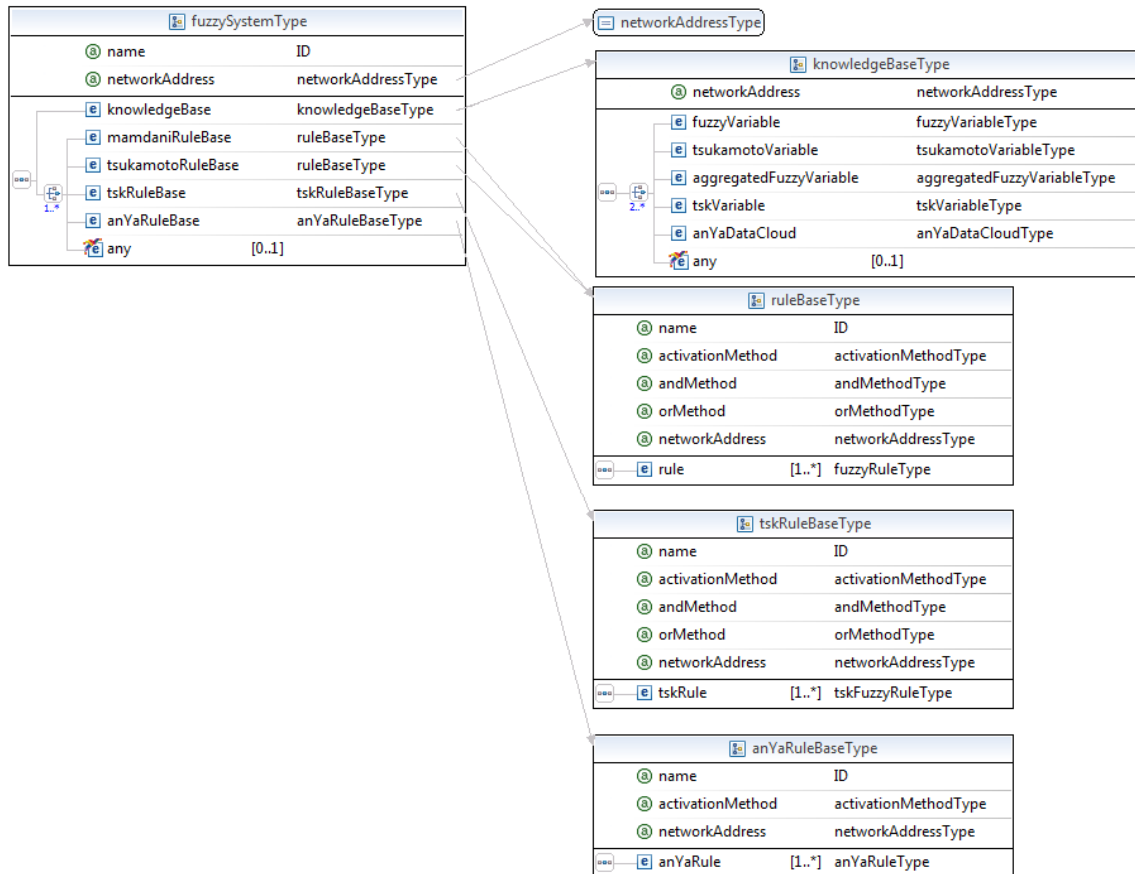


Figure 5—Type diagram for the elements *fuzzySystem*, *knowledgeBase*, *mamdaniRuleBase*, *tsukamotoRuleBase*, *tsKRuleBase*, and *anYaRuleBase*

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.

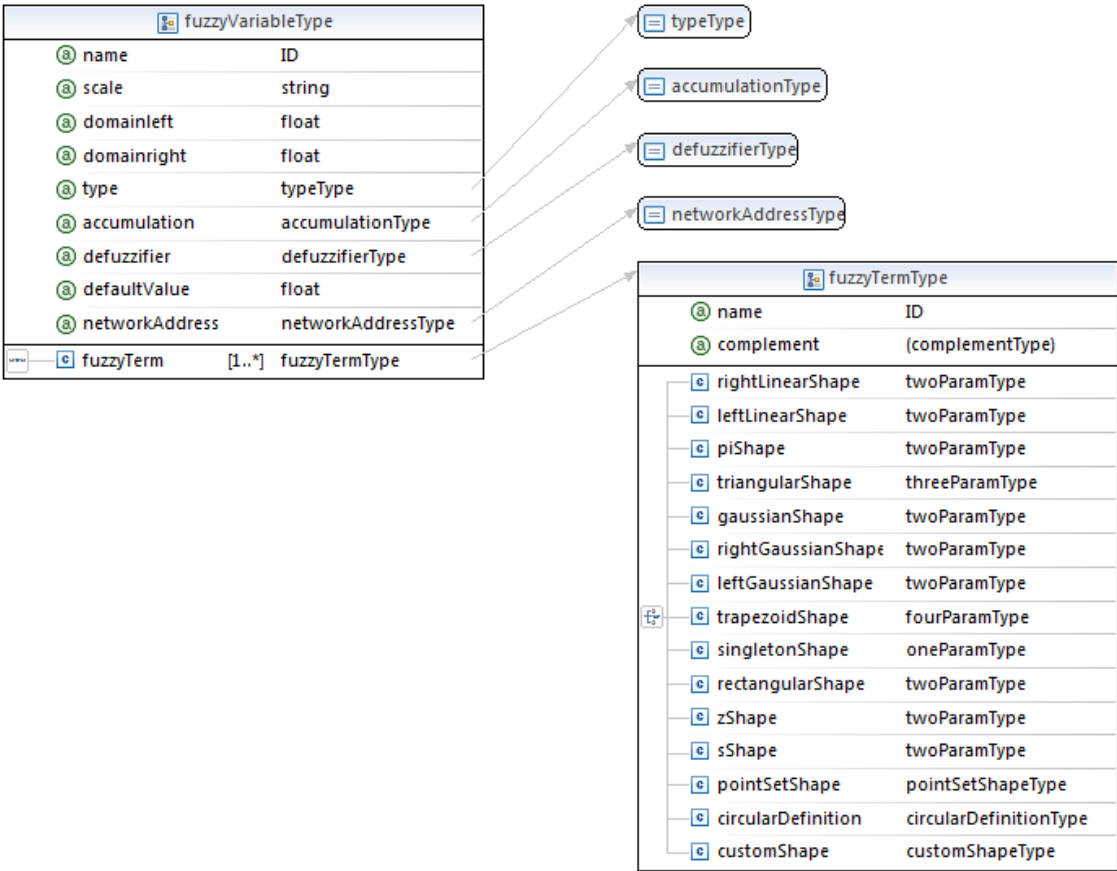


Figure 6—Type diagram for the element `fuzzyVariable`

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