

**NORME
INTERNATIONALE
INTERNATIONAL
STANDARD**

**CEI
IEC**

61131-7

Première édition
First edition
2000-08

Automates programmables –

**Partie 7:
Programmation en logique floue**

Programmable controllers –

**Part 7:
Fuzzy control programming**



Numéro de référence
Reference number
CEI/IEC 61131-7:2000

Numéros des publications

Depuis le 1er janvier 1997, les publications de la CEI sont numérotées à partir de 60000.

Publications consolidées

Les versions consolidées de certaines publications de la CEI incorporant les amendements sont disponibles. Par exemple, les numéros d'édition 1.0, 1.1 et 1.2 indiquent respectivement la publication de base, la publication de base incorporant l'amendement 1, et la publication de base incorporant les amendements 1 et 2.

Validité de la présente publication

Le contenu technique des publications de la CEI est constamment revu par la CEI afin qu'il reflète l'état actuel de la technique.

Des renseignements relatifs à la date de reconfirmation de la publication sont disponibles dans le Catalogue de la CEI.

Les renseignements relatifs à des questions à l'étude et des travaux en cours entrepris par le comité technique qui a établi cette publication, ainsi que la liste des publications établies, se trouvent dans les documents ci-dessous:

- «Site web» de la CEI*
- **Catalogue des publications de la CEI**
Publié annuellement et mis à jour régulièrement
(Catalogue en ligne)*
- **Bulletin de la CEI**
Disponible à la fois au «site web» de la CEI* et comme périodique imprimé

Terminologie, symboles graphiques et littéraux

En ce qui concerne la terminologie générale, le lecteur se reportera à la CEI 60050: *Vocabulaire Electrotechnique International* (VEI).

Pour les symboles graphiques, les symboles littéraux et les signes d'usage général approuvés par la CEI, le lecteur consultera la CEI 60027: *Symboles littéraux à utiliser en électrotechnique*, la CEI 60417: *Symboles graphiques utilisables sur le matériel. Index, relevé et compilation des feuilles individuelles*, et la CEI 60617: *Symboles graphiques pour schémas*.

* Voir adresse «site web» sur la page de titre.

Numbering

As from 1 January 1997 all IEC publications are issued with a designation in the 60000 series.

Consolidated publications

Consolidated versions of some IEC publications including amendments are available. For example, edition numbers 1.0, 1.1 and 1.2 refer, respectively, to the base publication, the base publication incorporating amendment 1 and the base publication incorporating amendments 1 and 2.

Validity of this publication

The technical content of IEC publications is kept under constant review by the IEC, thus ensuring that the content reflects current technology.

Information relating to the date of the reconfirmation of the publication is available in the IEC catalogue.

Information on the subjects under consideration and work in progress undertaken by the technical committee which has prepared this publication, as well as the list of publications issued, is to be found at the following IEC sources:

- **IEC web site***
- **Catalogue of IEC publications**
Published yearly with regular updates
(On-line catalogue)*
- **IEC Bulletin**
Available both at the IEC web site* and as a printed periodical

Terminology, graphical and letter symbols

For general terminology, readers are referred to IEC 60050: *International Electrotechnical Vocabulary* (IEV).

For graphical symbols, and letter symbols and signs approved by the IEC for general use, readers are referred to publications IEC 60027: *Letter symbols to be used in electrical technology*, IEC 60417: *Graphical symbols for use on equipment. Index, survey and compilation of the single sheets* and IEC 60617: *Graphical symbols for diagrams*.

* See web site address on title page.

**NORME
INTERNATIONALE
INTERNATIONAL
STANDARD**

**CEI
IEC**

61131-7

Première édition
First edition
2000-08

Automates programmables –

**Partie 7:
Programmation en logique floue**

Programmable controllers –

**Part 7:
Fuzzy control programming**

© IEC 2000 Droits de reproduction réservés — Copyright - all rights reserved

Aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de l'éditeur.

No part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from the publisher.

International Electrotechnical Commission
Telefax: +41 22 919 0300

3, rue de Varembé Geneva, Switzerland
e-mail: inmail@iec.ch IEC web site <http://www.iec.ch>



Commission Electrotechnique Internationale
International Electrotechnical Commission
Международная Электротехническая Комиссия

CODE PRIX
PRICE CODE

XA

Pour prix, voir catalogue en vigueur
For price, see current catalogue

SOMMAIRE

	Pages
AVANT-PROPOS	8
INTRODUCTION	12
Articles	
1 Domaine d'application et objet.....	16
2 Références normatives.....	16
3 Définitions.....	16
4 Intégration dans l'automate programmable	20
5 Langage de contrôle flou FCL.....	22
5.1 Echange de programmes de contrôle flou.....	22
5.2 Eléments du langage de contrôle flou	24
5.3 Exemple de FCL	42
5.4 Règles de production et mots-clés du langage de contrôle flou (FCL)	42
6 Conformité	50
6.1 Classes de conformité du langage de contrôle flou FCL.....	50
6.2 Liste de contrôle de données.....	54
Annexe A (informative) Théorie.....	58
A.1 Logique floue	58
A.2 Contrôle flou	66
A.3 Performances du contrôle flou.....	80
Annexe B (informative) Exemples	84
B.1 Précontrôle	84
B.2 Adaptation des paramètres d'automate PID conventionnel.....	86
B.3 Contrôle flou direct d'un procédé	86
Annexe C (informative) Exemple d'application industrielle – Grue à conteneurs.....	88
Annexe D (informative) Exemple d'utilisation de variables dans le bloc de règles.....	108
Annexe E (informative) Symboles, abréviations et synonymes.....	112
Figure 1 – Exemple de bloc fonction de contrôle flou en représentation FBD	22
Figure 2 – Echange de données entre programmes en langage de contrôle flou (FCL)	24
Figure 3 – Déclaration d'interface de bloc fonction en langage ST et FBD	26
Figure 4 – Exemple de termes de rampe.....	28
Figure 5 – Exemple d'utilisation de variables pour les fonctions d'appartenance	28
Figure 6 – Exemple de termes singletons.....	30
Figure 7 – Exemple de bloc fonction flou.....	42
Figure 8 – Niveaux de conformité.....	50
Figure A.1 – Fonctions d'appartenance pour les termes «âge adulte légal» et «adulte»	60
Figure A.2 – Description de la variable linguistique «Age» par des termes linguistiques et par leur hiérarchie sur l'échelle de temps (années d'âge)	60
Figure A.3 – Profils de fonctions d'appartenance fréquemment utilisés.....	62

CONTENTS

	Page
FOREWORD	9
INTRODUCTION	13
Clause	
1 Scope and object	17
2 Normative references	17
3 Definitions	17
4 Integration into the programmable controller	21
5 Fuzzy Control Language FCL	23
5.1 Exchange of fuzzy control programs	23
5.2 Fuzzy Control Language elements	25
5.3 FCL example	43
5.4 Production rules and keywords of the Fuzzy Control Language (FCL)	43
6 Compliance	51
6.1 Conformance classes of Fuzzy Control Language FCL	51
6.2 Data check list	55
Annex A (informative) Theory	59
A.1 Fuzzy Logic	59
A.2 Fuzzy Control	67
A.3 Performance of Fuzzy control	81
Annex B (informative) Examples	85
B.1 Pre-control	85
B.2 Parameter adaptation of conventional PID controllers	87
B.3 Direct fuzzy control of a process	87
Annex C (informative) Industrial example – Container crane	89
Annex D (informative) Example for using variables in the rule block	109
Annex E (informative) Symbols, abbreviations and synonyms	113
Figure 1 – Example of a fuzzy control Function Block in FBD representation	23
Figure 2 – Data exchange of Programs in Fuzzy Control Language (FCL)	25
Figure 3 – Example of a Function Block interface declaration in ST and FBD languages	27
Figure 4 – Example of ramp terms	29
Figure 5 – Example of usage of variables for membership functions	29
Figure 6 – Example of singleton terms	31
Figure 7 – Example for fuzzy function block	43
Figure 8 – Levels of conformance	51
Figure A.1 – Membership functions of the terms "full legal age" and "adult"	61
Figure A.2 – Description of the linguistic variable "Age" by linguistic terms and their hierarchy on the time scale (age in years)	61
Figure A.3 – Commonly used shapes of membership functions	63

Figure A.4 – Algorithmes pour l'implémentation d'opérations entre deux fonctions d'appartenance	66
Figure A.5 – Structure et éléments fonctionnels du contrôle flou.....	68
Figure A.6 – Principe de fuzzification (exemple)	68
Figure A.7 – Représentation de la base de connaissance sous forme linguistique	70
Figure A.8 – Représentation de deux variables sous forme de matrice	70
Figure A.9 – Eléments d'une inférence.....	72
Figure A.10a – Exemple montrant les principes d'agrégation.....	74
Figure A.10b – Principes d'activation (exemple)	74
Figure A.10c – Principes d'accumulation (exemple).....	76
Figure A.11a – Méthodes de défuzzification	76
Figure A.11b – Comparaison entre maximum le plus à gauche et maximum le plus à droite ..	78
Figure A.11c: Comparaison entre Centre de surface et Centre de gravité.....	78
Figure A.11d – Méthodes de défuzzification	80
Figure A.12 – Exemples de courbes de caractéristiques de contrôle flou	82
Figure A.13a – Automate à base de logique floue: structure fondamentale	82
Figure A.13b – Exemple d'automate à base de logique floue	82
Figure B.1 – Exemple de précontrôle	84
Figure B.2 – Exemple d'adaptation d'un paramètre.....	86
Figure B.3 – Exemple de contrôle flou direct	86
Figure C.1 – Exemple d'application industrielle – Grue à conteneurs	88
Figure C.2 – Variable linguistique «Distance» entre tête de grue et position finale	90
Figure C.3 – Variable linguistique «Angle» entre conteneur et tête de grue.....	90
Figure C.4 – Variable linguistique «Puissance»	90
Figure C.5 – Base de règles	92
Figure C.6 – Fuzzification de la variable linguistique «Distance»	92
Figure C.7 – Fuzzification de la variable linguistique «Angle»	94
Figure C.8 – Sous-ensemble de trois règles.....	94
Figure C.9 – Eléments d'agrégation	94
Figure C.10 – Principes d'agrégation	96
Figure C.11 – Eléments d'activation	96
Figure C.12 – Principes d'activation	98
Figure C.13 – Eléments d'accumulation	98
Figure C.14 – Principes d'accumulation	100
Figure C.15 – Défuzzification	102
Figure C.16 – Exemple en FCL	104
Figure D.1 – Principe du système contrôlé	108
Figure D.2 – Principe de contrôle flou d'un four	108
Figure D.3 – Bloc de règles.....	108
Figure D.4 – Exemple en FCL	110

	Page
Figure A.4 – Algorithms for implementing operations between two membership functions	67
Figure A.5 – Structure and functional elements of fuzzy control	69
Figure A.6 – The principle of fuzzification (as an example)	69
Figure A.7 – Representation of the knowledge base in linguistic form	71
Figure A.8 – Matrix representation of two variables	71
Figure A.9 – Elements of inference	73
Figure A.10a – An example showing the principles of aggregation	75
Figure A.10b – The principles of activation (as an example)	75
Figure A.10c – The principles of accumulation (as an example)	77
Figure A.11a – Methods of defuzzification	77
Figure A.11b – Difference between Left Most Maximum and Right Most Maximum	79
Figure A.11c – Difference between Centre of Area and Centre of Gravity	79
Figure A.11d – Methods of defuzzification	81
Figure A.12 – Examples of fuzzy control characteristic curves	83
Figure A.13a – Fuzzy-based controller: Fundamental structure	83
Figure A.13b – Example of a Fuzzy-based controller	83
Figure B.1 – Example of a pre-control	85
Figure B.2 – Example of a parameter adaptation	87
Figure B.3 – Example of a direct fuzzy control	87
Figure C.1 – Industrial example – Container crane	89
Figure C.2 – Linguistic variable "Distance" between crane head and target position	91
Figure C.3 – Linguistic variable "Angle" of the container to the crane head	91
Figure C.4 – Linguistic variable "Power"	91
Figure C.5 – Rule base	93
Figure C.6 – Fuzzification of the linguistic variable "distance"	93
Figure C.7 – Fuzzification of the linguistic variable "angle"	95
Figure C.8 – Subset of three rules	95
Figure C.9 – Elements of aggregation	95
Figure C.10 – Principles of aggregation	97
Figure C.11 – Elements of activation	97
Figure C.12 – Principles of activation	99
Figure C.13 – Elements of accumulation	99
Figure C.14 – Principles of accumulation	101
Figure C.15 – Defuzzification	103
Figure C.16 – Example in SCL	105
Figure D.1 – Principle of the controlled system	109
Figure D.2 – Principle of the fuzzy based control of the oven	109
Figure D.3 – Rule block	109
Figure D.4 – Example in FCL	111

	Pages
Tableau 1 – Méthodes de défuzzification.....	30
Tableau 2 – Formules des différentes méthodes de défuzzification	32
Tableau 3 – Algorithmes appariés.....	34
Tableau 4 – Méthodes d'activation	34
Tableau 5 – Méthodes d'accumulation	36
Tableau 6 – Priorité des opérateurs	36
Tableau 7 – Mots-clés réservés pour le FCL	48
Tableau 8 – Eléments de langage de contrôle flou FCL Niveau Base (obligatoires).....	52
Tableau 9 – Eléments de langage de contrôle flou FCL Niveau Extension (facultatifs).....	54
Tableau 10 – Exemple de liste d'éléments de langage Niveau Ouvert.....	54
Tableau 11 – Liste de contrôle de données	56
Tableau A.1 – Etapes d'inférence et algorithmes couramment utilisés	74
Tableau C.1 – Etapes d'inférence et opérateurs correspondants	92
Tableau E.1 – Symboles et Abréviations	112
Tableau E.2 – Synonymes	112

IECNORM.COM : Click to view the full PDF of IEC 61131-7:2000

	Page
Table 1 – Defuzzification methods	31
Table 2 – Formulae for defuzzification methods	33
Table 3 – Paired algorithms	35
Table 4 – Activation methods	35
Table 5 – Accumulation methods	37
Table 6 – Priority of operators	37
Table 7 – Reserved keywords for FCL	49
Table 8 – FCL Basic Level language elements (mandatory).....	53
Table 9 – FCL Extension Level language elements (optional).....	55
Table 10 – Examples of a list with Open Level language elements	55
Table 11 – Data check list	57
Table A.1 – Inference steps and commonly used algorithms.....	75
Table C.1 – Inference steps and assigned operator.....	93
Table E.1 – Symbols and abbreviations.....	113
Table E.2 – Synonyms	113

IECNORM.COM : Click to view the full PDF of IEC 61131-7:2000

COMMISSION ÉLECTROTECHNIQUE INTERNATIONALE

AUTOMATES PROGRAMMABLES –

Partie 7: Programmation en logique floue

AVANT-PROPOS

- 1) La CEI (Commission Électrotechnique Internationale) est une organisation mondiale de normalisation composée de l'ensemble des comités électrotechniques nationaux (Comités nationaux de la CEI). La CEI a pour objet de favoriser la coopération internationale pour toutes les questions de normalisation dans les domaines de l'électricité et de l'électronique. A cet effet, la CEI, entre autres activités, publie des Normes internationales. Leur élaboration est confiée à des comités d'études, aux travaux desquels tout Comité national intéressé par le sujet traité peut participer. Les organisations internationales, gouvernementales et non gouvernementales, en liaison avec la CEI, participent également aux travaux. La CEI collabore étroitement avec l'Organisation Internationale de Normalisation (ISO), selon des conditions fixées par accord entre les deux organisations.
- 2) Les décisions ou accords officiels de la CEI concernant les questions techniques représentent, dans la mesure du possible, un accord international sur les sujets étudiés, étant donné que les Comités nationaux intéressés sont représentés dans chaque comité d'études.
- 3) Les documents produits se présentent sous la forme de recommandations internationales. Ils sont publiés comme normes, spécifications techniques, rapports techniques ou guides et agréés comme tels par les Comités nationaux.
- 4) Dans le but d'encourager l'unification internationale, les Comités nationaux de la CEI s'engagent à appliquer de façon transparente, dans toute la mesure possible, les Normes internationales de la CEI dans leurs normes nationales et régionales. Toute divergence entre la norme de la CEI et la norme nationale ou régionale correspondante doit être indiquée en termes clairs dans cette dernière.
- 5) La CEI n'a fixé aucune procédure concernant le marquage comme indication d'approbation et sa responsabilité n'est pas engagée quand un matériel est déclaré conforme à l'une de ses normes.
- 6) L'attention est attirée sur le fait que certains des éléments de la présente Norme internationale peuvent faire l'objet de droits de propriété intellectuelle ou de droits analogues. La CEI ne saurait être tenue pour responsable de ne pas avoir identifié de tels droits de propriété et de ne pas avoir signalé leur existence.

La Norme internationale CEI 61131-7 a été établie par le sous-comité 65B: Dispositifs, du comité d'études 65 de la CEI: Mesure et commande dans les processus industriels.

Le texte de cette norme est issu des documents suivants:

FDIS	Rapport de vote
65B/406/FDIS	65B/413/RVD

Le rapport de vote indiqué au tableau ci-dessus donne toute information sur le vote ayant abouti à l'approbation de cette norme.

Cette publication a été rédigée selon les Directives ISO/CEI, Partie 3.

Les annexes A à E sont données uniquement à titre d'information.

La CEI 61131 comprend les parties suivantes, présentées sous le titre général *Automates programmables*:

Partie 1:1992, Informations générales

Partie 2:1992, Spécifications et essais des équipements

Partie 3:1993, Langages de programmation

INTERNATIONAL ELECTROTECHNICAL COMMISSION

PROGRAMMABLE CONTROLLERS –**Part 7: Fuzzy control programming****FOREWORD**

- 1) The IEC (International Electrotechnical Commission) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of the IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, the IEC publishes International Standards. Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. The IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
- 2) The formal decisions or agreements of the IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested National Committees.
- 3) The documents produced have the form of recommendations for international use and are published in the form of standards, technical specifications, technical reports or guides and they are accepted by the National Committees in that sense.
- 4) In order to promote international unification, IEC National Committees undertake to apply IEC International Standards transparently to the maximum extent possible in their national and regional standards. Any divergence between the IEC Standard and the corresponding national or regional standard shall be clearly indicated in the latter.
- 5) The IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with one of its standards.
- 6) Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights. The IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC 61131-7 has been prepared by subcommittee 65B: Devices, of IEC technical committee 65: Industrial-process measurement and control.

The text of this standard is based on the following documents:

FDIS	Report on voting
65B/406/FDIS	65B/413/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives, Part 3.

Annexes A to E are for information only.

IEC 61131 consists of the following parts under the general title *Programmable controllers*:

Part 1:1992, General information

Part 2:1992, Equipment requirements and tests

Part 3:1993, Programming languages

Partie 4:1995, Guide pour l'utilisateur

Partie 5: Communication (à publier)

Partie 6: Communications pour automates programmables par le bus de terrain (à l'étude)

Partie 7: Programmation en logique floue

Le comité a décidé que le contenu de cette publication ne sera pas modifié avant 2005. A cette date, la publication sera

- reconduite;
- supprimée;
- remplacée par une édition révisée, ou
- amendée.

IECNORM.COM : Click to view the full PDF of IEC 61131-7:2000

Part 4:1995, User guidelines

Part 5: Communications (to be published)

Part 6: Programmable controller communications via fieldbus (under study)

Part 7: Fuzzy control programming

The committee has decided that the contents of this publication will remain unchanged until 2005. At this date, the publication will be

- reconfirmed;
- withdrawn;
- replaced by a revised edition, or
- amended.

IECNORM.COM : Click to view the full PDF of IEC 61131-7:2000

INTRODUCTION

La logique floue appliquée au contrôle est connue sous le nom de «contrôle flou». Le contrôle flou se présente comme une technologie capable d'augmenter les capacités des automatismes de contrôle industriel, et est adaptée aux tâches de niveau de contrôle généralement prises en charge par des automates programmables.

Le contrôle flou se base sur une connaissance pratique représentée par des bases de règles linguistiques, et non par des modèles analytiques (empiriques ou théoriques). Son utilisation est possible lorsqu'un savoir-faire est transcriptible dans sa formalisation. Il permet alors d'exploiter les connaissances acquises sur le terrain pour améliorer les processus et exécuter différentes tâches, comme par exemple

- contrôle (boucle fermée ou ouverte, à une variable ou plusieurs variables, pour systèmes linéaires ou non linéaires),
- réglage en ligne ou hors-ligne de paramètres de systèmes de contrôle,
- classification et reconnaissance des formes,
- prise de décision en temps réel (produit envoyé vers machine A ou B ?),
- aide à l'opérateur humain pour la prise de décision ou le réglage de paramètres,
- détection et diagnostic de défaillances de systèmes.

Un éventail d'applications très variées et une approche naturelle inspirée de l'expérience humaine ont fait du contrôle flou un outil essentiel, destiné à devenir un standard accessible à l'ensemble des utilisateurs d'automates programmables.

Il est également possible de combiner le contrôle flou à des méthodes de contrôle classiques, et ce sans difficulté particulière.

L'application du contrôle flou est avantageuse dans les cas où aucun modèle explicite du processus n'est disponible, ou lorsque le modèle analytique est trop difficile à évaluer ou trop compliqué pour être évalué en temps réel.

Le contrôle flou présente également l'avantage d'incorporer l'expérience humaine de manière simple et directe. Par ailleurs, le contrôle flou ne nécessite qu'une modélisation partielle de l'automate: parfois par simple interpolation entre plusieurs modèles localement linéaires ou par adaptation dynamique des paramètres d'un «automate linéaire», le rendant ainsi non linéaire, ou bien sur un automate existant, en agissant sur une fonctionnalité spécifique pour l'améliorer.

Le contrôle flou est un contrôle à valeurs multiples; la proposition de contrôle n'est donc plus limitée aux deux valeurs «vrai» ou «faux». Cette caractéristique du contrôle flou le rend particulièrement utile pour modéliser le savoir-faire empirique et spécifier les actions de contrôle à prendre sur un ensemble donné de valeurs d'entrées.

Sur la théorie existante et les systèmes déjà réalisés en contrôle flou, on observe des écarts importants en matière de terminologie (définitions), de caractéristiques (fonctionnalités) et de mise en œuvre (outils).

Le contrôle flou se retrouve dans des applications simples et modestes tout comme sur des projets hautement sophistiqués et complexes. Pour assurer une grande variété d'applications à la présente partie de la CEI 61131, les caractéristiques d'un système de contrôle flou conforme sont réparties en différentes classes de conformité.

La classe de base définit l'ensemble de caractéristiques minimal que tout système conforme doit avoir. Cette exigence facilite l'échange de programmes de contrôle flou.

INTRODUCTION

The theory of fuzzy logic in the application of control is named fuzzy control. Fuzzy control is emerging as a technology that can enhance the capabilities of industrial automation, and is suitable for control level tasks generally performed in Programmable Controllers (PC).

Fuzzy control is based upon practical application knowledge represented by so-called linguistic rule bases, rather than by analytical (either empirical or theoretical) models. Fuzzy control can be used when there is an expertise that can be expressed in its formalism. That allows to take available knowledge to improve processes and perform a variety of tasks, for instance

- control (closed or open loop, single or multi-variable, for linear or non-linear systems),
- on-line or off-line setting of control systems' parameters,
- classification and pattern recognition,
- real-time decision making (send this product to machine A or B ?),
- helping operators to make decisions or tune parameters,
- detection and diagnosis of faults in systems.

Its wide range of applications and natural approach based on human experience makes fuzzy control a basic tool that should be made available to programmable controller users as a standard.

Fuzzy control can also, in a straightforward way, be combined with classical control methods.

The application of fuzzy control can be of advantage in such cases where there is no explicit process model available, or in which the analytical model is too difficult to evaluate or when the model is too complicated to evaluate in real time.

Another advantageous feature of fuzzy control is that human experience can be incorporated in a straightforward way. Also, it is not necessary to model the whole controller with fuzzy control: sometimes fuzzy control just interpolates between a series of locally linear models, or dynamically adapts the parameters of a "linear controller", thereby rendering it non-linear, or alternatively just "zoom in" onto a certain feature of an existing controller that needs to be improved.

Fuzzy control is a multi-valued control, no longer restricting the values of a control proposition to "true" or "false". This makes fuzzy control particularly useful to model empirical expertise, stating which control actions have to be taken under a given set of inputs.

The existing theory and systems already realized in the area of fuzzy control differ widely in terms of terminology (definitions), features (functionalities) and implementation (tools).

Fuzzy control is used from small and simple applications up to highly sophisticated and complex projects. To cover all kinds of usage in this part of IEC 61131, the features of a compliant fuzzy control system are mapped into defined conformance classes.

The basic class defines a minimum set of features which has to be achieved by all compliant systems. This facilitates the exchange of fuzzy control programs.

Des caractéristiques facultatives de la norme sont définies dans la classe d'extension. Les programmes de contrôle flou qui appliquent ces caractéristiques ne sont entièrement portés qu'entre des systèmes basés sur le même ensemble de caractéristiques; la portabilité n'est que partielle dans les autres cas. La présente Norme n'oblige pas chaque système conforme à présenter toutes les caractéristiques de la classe d'extension, mais elle prévoit la possibilité de portabilité (partielle) et décourage l'utilisation de caractéristiques non standard. Par conséquent, il convient qu'un système conforme n'offre pas de caractéristiques non standard pouvant être correctement réalisées à partir des caractéristiques standard de la classe de base et de la classe d'extension.

Pour éviter que les systèmes eux-mêmes pourvus de caractéristiques hautement sophistiquées ne puissent répondre aux exigences de la présente partie de la CEI 61131 et pour ne pas gêner le progrès des évolutions à venir, cette norme permet également d'intégrer des caractéristiques non standard supplémentaires qui ne sont couvertes ni par la classe de base ni par la classe d'extension. Cependant, une présentation standard de ces caractéristiques est nécessaire afin de facilement les identifier comme des caractéristiques non standard.

La portabilité des applications de contrôle flou varie suivant les différents systèmes de programmation, et dépend également des caractéristiques des systèmes de contrôle. Ces dépendances sont couvertes par la liste de contrôle de données qui doit être fournie par le fabricant.

IECNORM.COM : Click to view the full PDF of IEC 61131-7:2000

Optional standard features are defined in the extension class. Fuzzy control programs applying these features can only be fully ported among systems using the same set of features, otherwise a partial exchange may be possible only. This standard does not force all compliant systems to realize all features in the extension class, but it supports the possibility of (partial) portability and the avoidance of the usage of non-standard features. Therefore, a compliant system should not offer non-standard features which can be meaningfully realized by using standard features of the basic class and the extension class.

In order not to exclude systems using their own highly sophisticated features from complying with this part of IEC 61131 and not to hinder the progress of future development, this standard permits also additional non-standard features which are not covered by the basic class and the extension class. However, these features need to be listed in a standard way to ensure that they are easily recognised as non-standard features.

The portability of fuzzy control applications depends on the different programming systems and also the characteristics of the control systems. These dependencies are covered by the data check list to be delivered by the manufacturer.

IECNORM.COM : Click to view the full PDF of IEC 61131-7:2000

AUTOMATES PROGRAMMABLES –

Partie 7: Programmation en logique floue

1 Domaine d'application et objet

La présente partie de la CEI 61131 définit un langage permettant de programmer des applications de contrôle flou pour automates programmables.

L'objet de cette partie de la CEI 61131 est de fournir aux fabricants et aux utilisateurs une définition commune et précise des principales méthodes d'intégration des applications de contrôle flou dans les langages d'automates programmables, selon la CEI 61131-3, et de permettre une portabilité des programmes de contrôle flou entre différents systèmes de programmation.

Dans cette optique, l'annexe A présente une introduction aux théories de contrôle flou, brève mais suffisante pour aider à mieux comprendre la présente partie de la CEI 61131. Les lecteurs de cette partie de la CEI 61131 non familiarisés avec la théorie de contrôle flou sont invités à commencer par l'annexe A.

2 Références normatives

Les documents normatifs suivants contiennent des dispositions qui, par suite de la référence qui y est faite, constituent des dispositions valables pour la présente partie de la CEI 61131. Pour les références datées, les amendements ultérieurs ou les révisions de ces publications ne s'appliquent pas. Toutefois, les parties prenantes aux accords fondés sur la présente partie de la CEI 61131 sont invitées à rechercher la possibilité d'appliquer les éditions les plus récentes des documents normatifs indiqués ci-après. Pour les références non datées, la dernière édition du document normatif en référence s'applique. Les membres de la CEI et de l'ISO possèdent le registre des Normes internationales en vigueur.

CEI 60050-351:1998, *Vocabulaire Electrotechnique International (VEI) – Partie 351: Commande et régulation automatiques*

CEI 61131-3:1993, *Automates programmables – Partie 3: Langages de programmation*

3 Définitions

Pour les besoins de la présente partie de la CEI 61131, les définitions suivantes s'appliquent.

Des définitions supplémentaires d'éléments de langage sont présentées dans la CEI 61131-3.

NOTE Les termes définis dans cet article sont indiqués en italique lorsqu'ils apparaissent dans le corps des définitions.

3.1

accumulation

combinaison des résultats de *règles linguistiques* dans un résultat final

3.2

agrégation

calcul du degré de vérité de la *condition* d'une règle

combinaison des degrés d'appartenance de toutes les sous-conditions individuelles dans une règle de calcul du degré d'accomplissement de la *condition* d'une règle

PROGRAMMABLE CONTROLLERS –

Part 7: Fuzzy control programming

1 Scope and object

This part of IEC 61131 defines a language for the programming of Fuzzy Control applications used by programmable controllers.

The object of this part of IEC 61131 is to offer the manufacturers and the users a well-defined common understanding of the basic means to integrate fuzzy control applications in the Programmable Controller languages according to IEC 61131-3, as well as the possibility to exchange portable fuzzy control programs among different programming systems.

To achieve this, annex A gives a short introduction to the theory of fuzzy control and fuzzy logic as far as it is necessary for the understanding of this part of IEC 61131. It may be helpful for readers of this part of IEC 61131 who are not familiar with fuzzy control theory to read annex A first.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of IEC 61131. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of IEC 61131 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of IEC and ISO maintain registers of currently valid International Standards.

IEC 60050-351:1998, *International Electrotechnical Vocabulary (IEV) – Part 351: Automatic control*

IEC 61131-3:1993, *Programmable controllers – Part 3: Programming languages*

3 Definitions

For the purpose of this part of IEC 61131, the following definitions apply.

Further definitions for language elements are given in IEC 61131-3.

NOTE Terms defined in this clause are italicized where they appear in the text of definitions.

3.1

accumulation

result aggregation

combination of results of *linguistic rules* in a final result

3.2

aggregation

determination of degree of firing

combination of membership degrees of all individual subconditions in a rule to calculation of the degree of accomplishment of the *condition* of a rule

3.3

activation

processus par lequel le degré de vérité d'une *condition* agit sur un ensemble flou de sortie

3.4

conclusion

conséquence

résultat d'une *règle linguistique*, qui définit les actions à prendre (la partie ALORS d'une règle de contrôle flou SI..ALORS)

3.5

condition

prédicat

expression composée de *sous-conditions* combinées par des *opérateurs flous* ET, OU, PAS

3.6

ensemble «net»

cas particulier d'*ensemble flou*, pour lequel la *fonction d'appartenance* ne prend que deux valeurs, généralement définies comme 0 et 1

3.7

défuzzification

conversion d'un *ensemble flou* en valeur numérique

3.8

degré d'appartenance

valeur de la fonction d'appartenance

3.9

fuzzification

calcul des *degrés d'appartenance* de la valeur d'entrée «nette» des *termes linguistiques* définis avec chaque *variable linguistique* d'entrée

3.10

contrôle flou

type de contrôle dont l'algorithme est basé sur la *logique floue*

[VEI 351-17-51, modifié]

3.11

logique floue

ensemble de théories mathématiques basées sur le concept d'*ensemble flou*. La *logique floue* est une sorte de logique à valeurs infinies

3.12

opérateur flou

opérateur utilisé dans la théorie de *logique floue*

3.13

ensemble flou

un *ensemble flou* A est défini comme l'ensemble de paires ordonnées $(x, \mu_A(x))$, où x est un élément de l'univers de discours U et $\mu_A(x)$ est la *fonction d'appartenance*. Celle-ci attribue à chaque élément $x \in U$ un nombre réel de l'intervalle [0,1] qui précise dans quelle mesure x appartient à l'ensemble

3.3 activation

process by which the degree of fulfilment of a *condition* acts on an output fuzzy set

3.4 conclusion consequent

output of a *linguistic rule*, i.e. the actions to be taken (the THEN part of an IF..THEN fuzzy control rule)

3.5 condition antecedent

expression comprising *subconditions* combined with *fuzzy operators* AND, OR, NOT

3.6 crisp set

special case of a *fuzzy set*, in which the *membership function* only takes two values, commonly defined as 0 and 1

3.7 defuzzification

conversion of a *fuzzy set* into a numerical value

3.8 degree of membership membership function value

3.9 fuzzification

determination of *degrees of membership* of the crisp input value of the *linguistic terms* defined with each input *linguistic variable*

3.10 fuzzy control

type of control in which the control algorithm is based on *fuzzy logic*

[IEV 351-17-51, modified]

3.11 fuzzy logic

collection of mathematical theories based on the notion of *fuzzy set*. *Fuzzy logic* is a kind of infinite-valued logic

3.12 fuzzy operator

operator used in *fuzzy logic* theory

3.13 fuzzy set

A *fuzzy set* A is defined as the set of ordered pairs $(x, \mu_A(x))$, where x is an element of the universe of discourse U and $\mu_A(x)$ is the *membership function*, that attributes to each $x \in U$ a real number $\in [0,1]$, describing the degree to which x belongs to the set

3.14

inférence

application de *règles linguistiques* sur des valeurs d'entrée dans le but de générer des valeurs de sortie

3.15

règle linguistique

règle de type SI-ALORS composée d'une *condition* et d'une *conclusion*, l'une ou l'autre ou toutes deux pouvant être linguistiques

3.16

terme linguistique

dans un contexte de contrôle flou, les *termes linguistiques* sont définis par des *ensembles flous*

3.17

variable linguistique

variable qui prend des valeurs comprises dans l'intervalle des *termes linguistiques*

3.18

fonction d'appartenance

fonction qui définit le *degré d'appartenance* à l'univers de discours pour un *ensemble flou* donné

[VEI 351-17-52, modifié]

3.19

singleton

ensemble flou dont la *fonction d'appartenance* sur un point est égale à *un*, et à *zéro* sur tous les autres points

3.20

sous-condition

expression élémentaire qui se présente comme une variable ou comme l'expression «*variable linguistique EST terme linguistique*»

3.21

base de règles

ensemble de *règles linguistiques* dont le rôle est d'atteindre certains objectifs

3.22

facteur de pondération

valeur comprise entre 0 et 1, qui précise le degré d'importance, de crédibilité et d'assurance d'une *règle linguistique*

4 Intégration dans l'automate programmable

Les applications de contrôle flou programmées en langage de contrôle flou FCL selon l'article 5 doivent être encapsulées dans des blocs fonction (ou programmes) tels que définis dans la CEI 61131-3. Le concept des types de bloc fonction et des instances de bloc fonction décrit dans la CEI 61131-3 s'applique à la présente norme.

Les types de bloc fonction définis en langage de contrôle flou FCL doivent spécifier les paramètres d'entrée et sortie, ainsi que les règles et déclarations spécifiques de contrôle flou.

Les instances de blocs fonction correspondantes doivent contenir les données spécifiques des applications de contrôle flou.

3.14**inference**

application of *linguistic rules* on input values in order to generate output values

3.15**linguistic rule**

IF-THEN rule with *condition* and *conclusion*, one or both linguistic

3.16**linguistic term**

in the context of fuzzy control, *linguistic terms* are defined by *fuzzy sets*

3.17**linguistic variable**

variable that takes values in the range of *linguistic terms*

3.18**membership function**

function which defines the *degree of membership* over the universe of discourse for a given *fuzzy set*

[IEV 351-17-52, modified]

3.19**singleton**

fuzzy set whose *membership function* is equal to one at one point and equal to zero at all other points

3.20**subcondition**

elementary expression in the form of a variable or as term "*linguistic variable IS linguistic term*"

3.21**rule base**

collection of *linguistic rules* to attain certain objectives

3.22**weighting factor**

value between 0..1, that states the degree of importance, credibility, confidence of a *linguistic rule*

4 Integration into the programmable controller

The fuzzy control applications programmed in Fuzzy Control Language FCL according to clause 5 shall be encapsulated in Function Blocks (or Programs) as defined in IEC 61131-3. The concept of Function Block Types and Function Block Instances given in IEC 61131-3 applies to this standard.

The Function Block Types defined in Fuzzy Control Language FCL shall specify the input and output parameters and the fuzzy control specific rules and declarations.

The corresponding Function Block Instances shall contain the specific data of the fuzzy control applications.

Les blocs fonction définis en langage de contrôle flou FCL peuvent être utilisés dans des programmes ou des blocs fonction écrits dans un des langages de la CEI 61131-3, par diagramme en échelle (ladder) ou liste d'instructions, par exemple. Les types de données des paramètres d'entrée et sortie du bloc fonction ou programme écrit en FCL doivent être identiques aux types de données correspondants de «l'environnement d'appel», comme illustré à la figure 1.

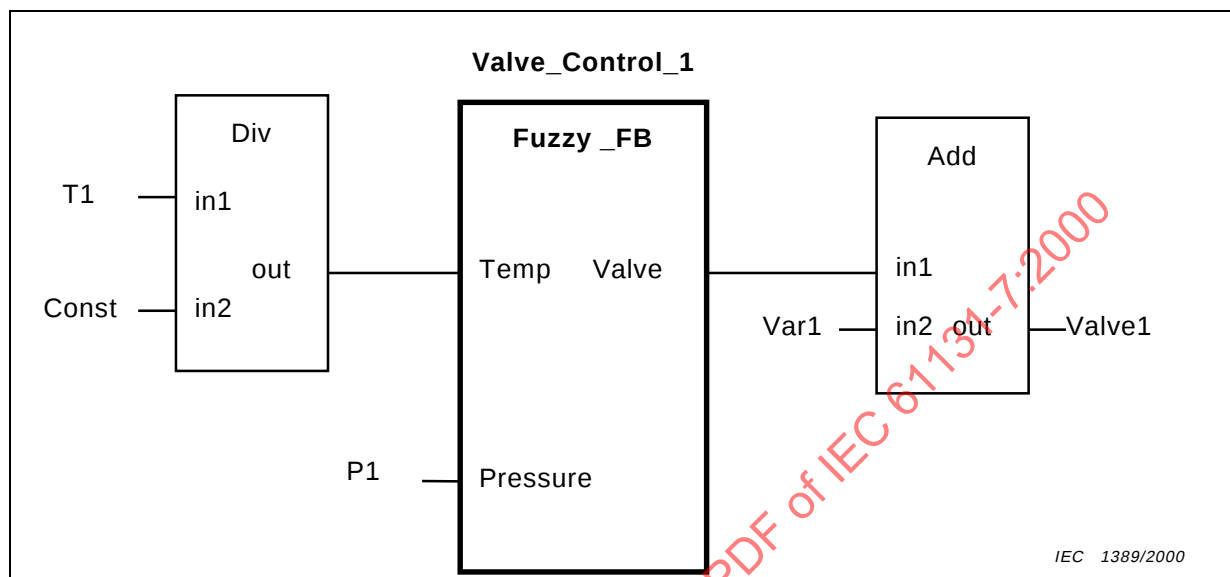


Figure 1 – Exemple de bloc fonction de contrôle flou en représentation FBD

Dans cet exemple, Valve_Control_1 est une instance de bloc fonction de type Fuzzy_FB. Le type de bloc fonction Fuzzy_FB peut être programmé en langage de contrôle flou FCL selon l'article 5. Le bloc fonction Fuzzy_FB est utilisé ici dans un programme ou un bloc fonction représenté dans le langage graphique FBD (diagramme de bloc fonction) de la CEI 61131-3.

5 Langage de contrôle flou FCL

5.1 Echange de programmes de contrôle flou

La définition du langage de contrôle flou FCL est basée sur les définitions des langages de programmation de la CEI 61131-3. L'interaction entre l'algorithme de contrôle flou et l'environnement de son programme a pour conséquence que l'algorithme est «caché» du programme. L'algorithme de contrôle flou est donc extérieurement représenté comme un bloc fonction selon la CEI 61131-3. Les éléments nécessaires pour décrire les parties linguistiques internes du bloc fonction de contrôle flou, tels que les fonctions d'appartenance, les règles, les opérateurs et les méthodes doivent être définis selon le présent article.

Les éléments de langage du FCL permettent une représentation commune pour l'échange de données entre différents outils de configuration de contrôle flou fournis par différents fabricants, comme illustré à la figure 2. Grâce à cette représentation commune, chaque fabricant d'automate programmable peut conserver son matériel, ses éditeurs de logiciel et ses compilateurs; il lui suffit d'implémenter l'interface de données dans son propre éditeur. Son client peut alors déplacer des projets de contrôle flou entre différents fabricants.

Function Blocks defined in Fuzzy Control Language FCL may be used in Programs and Function Blocks written in any of the languages of IEC 61131-3, for example, Ladder Diagram, Instruction List, etc. The data types of the input and output parameters of the Function Block or Program written in FCL shall match those of the corresponding "calling environment" as illustrated in figure 1.

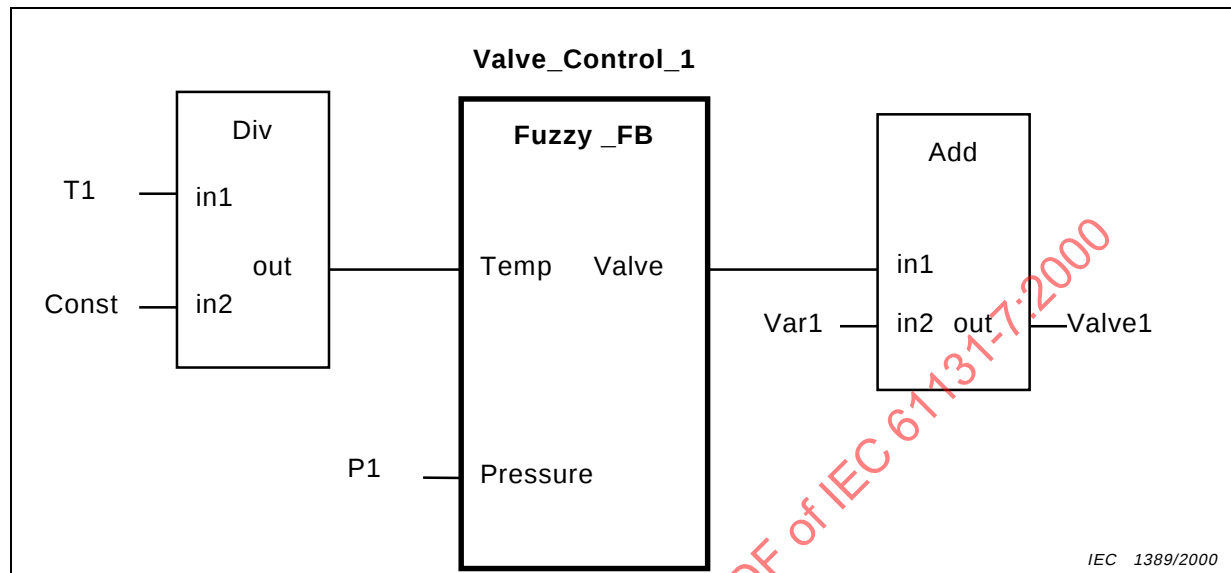


Figure 1 – Example of a fuzzy control Function Block in FBD representation

In this example, Valve_Control_1 is a user defined Function Block Instance of the Function Block Type Fuzzy_FB. The Function Block Type Fuzzy_FB may be programmed in Fuzzy Control Language FCL according to clause 5. The Function Block Fuzzy_FB is used here in a program or a Function Block which is represented in the graphical language FBD (Function Block Diagram) of IEC 61131-3.

5 Fuzzy Control Language FCL

5.1 Exchange of fuzzy control programs

The definition of the Fuzzy Control Language FCL is based on the definitions of the programming languages in IEC 61131-3. The interaction of the fuzzy control algorithm with its program environment causes it to be "hidden" from the program. The fuzzy control algorithm is therefore externally represented as a Function Block according to IEC 61131-3. The necessary elements for describing the internal linguistic parts of the fuzzy control Function Block like membership functions, rules, operators and methods have to be defined according to this clause.

The language elements of FCL standardize a common representation for data exchange among fuzzy control configuration tools of different manufacturers shown in figure 2. Using this common representation, every manufacturer of programmable controllers may keep his hardware, software editors and compilers. The manufacturer has only to implement the data interface into his specific editor. The customer would be able to exchange fuzzy control projects between different manufacturers.

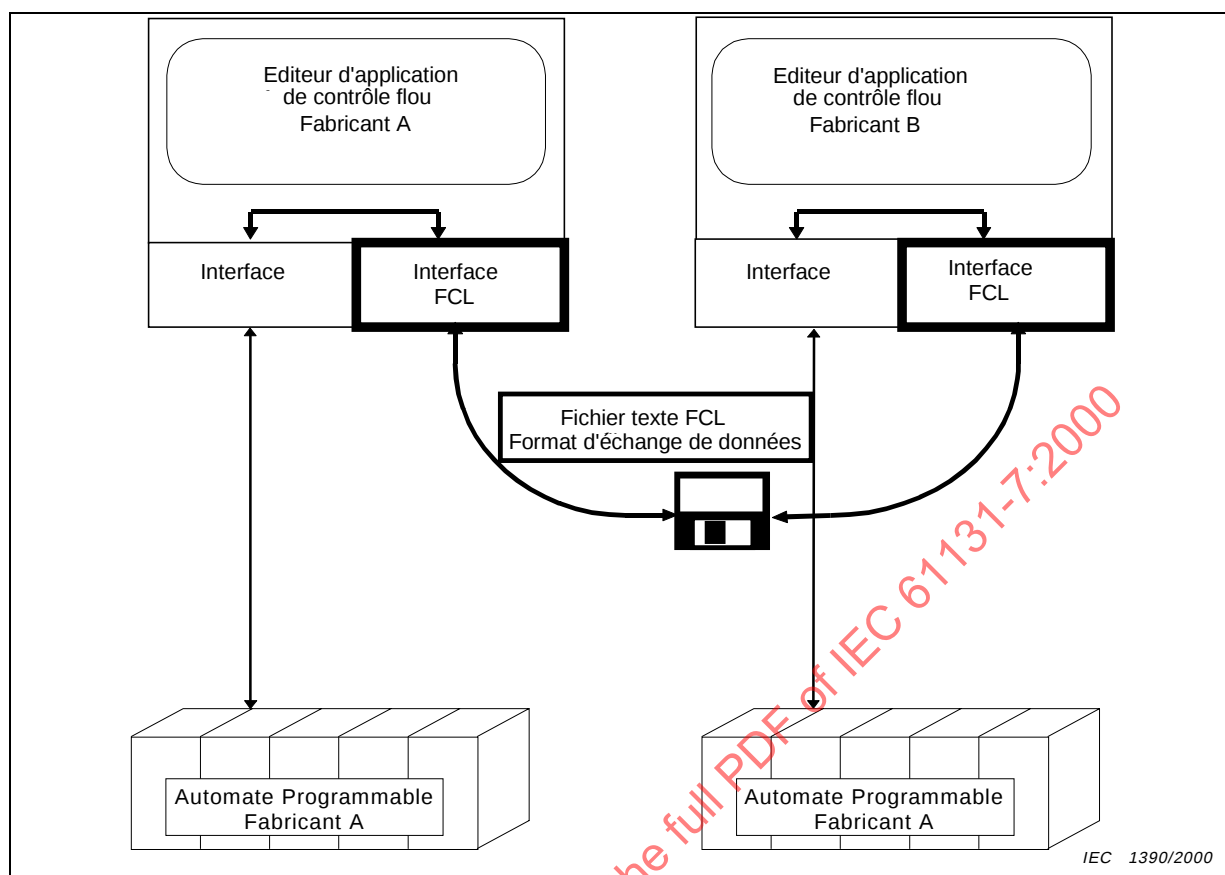


Figure 2 – Echange de données entre programmes en langage de contrôle flou (FCL)

5.2 Eléments du langage de contrôle flou

Les éléments du langage de contrôle flou du présent article sont décrits par des exemples. La règle de production détaillée est définie en 5.4.

5.2.1 Interface de bloc fonction

Selon l'article 4, la vue externe du bloc fonction flou fait appel aux éléments de langage standard de la CEI 61131-3 suivants:

FUNCTION_BLOCK <i>function_block_name</i>	Bloc fonction
VAR_INPUT	Déclaration de paramètre d'entrée
<i>variable_name: data_type;</i>	
....	
END_VAR	
VAR_OUTPUT	Déclaration de paramètre de sortie
<i>variable_name: data_type;</i>	
....	
END_VAR	
....	
VAR	Variables locales
<i>variable_name: data_type;</i>	
END_VAR	
END_FUNCTION_BLOCK	

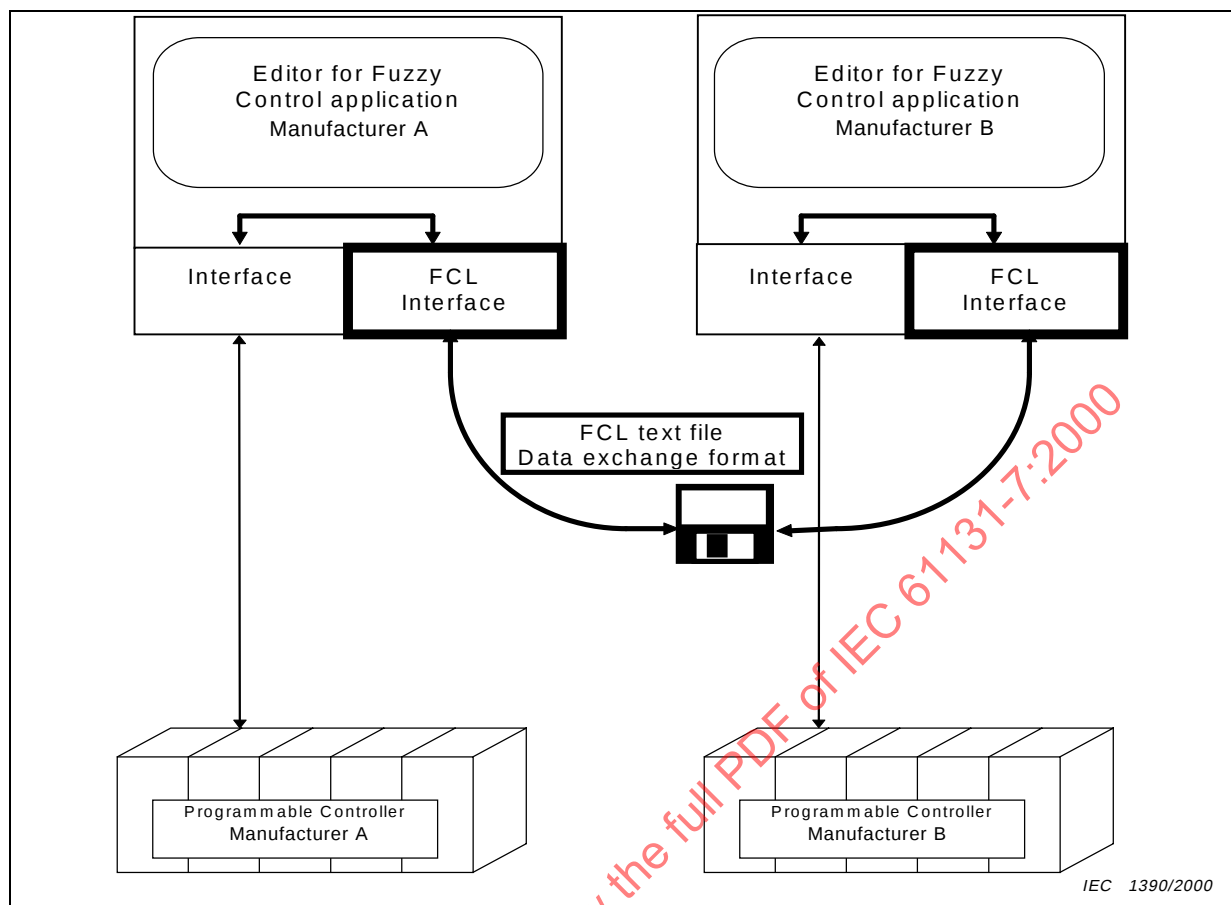


Figure 2 – Data exchange of Programs in Fuzzy Control Language (FCL)

5.2 Fuzzy Control Language elements

Fuzzy control language elements in this subclause are described using examples. The detailed production rule is given in 5.4.

5.2.1 Function Block interface

According to clause 4, the external view of the fuzzy Function Block requires that the following standard language elements of IEC 61131-3 be used:

<code>FUNCTION_BLOCK</code> <i>function_block_name</i>	Function block
<code>VAR_INPUT</code>	Input parameter declaration
<i>variable_name: data_type;</i>	
<code>....</code>	
<code>END_VAR</code>	
<code>VAR_OUTPUT</code>	Output parameter declaration
<i>variable_name: data_type;</i>	
<code>....</code>	
<code>END_VAR</code>	
<code>....</code>	
<code>VAR</code>	Local variables
<i>variable_name: data_type;</i>	
<code>END_VAR</code>	
<code>END_FUNCTION_BLOCK</code>	

Au moyen de ces éléments de langage, il est possible de décrire une interface de bloc fonction. Une interface de bloc fonction est définie par des paramètres qui entrent et sortent du bloc fonction. Les types de données de ces paramètres doivent être définis selon la CEI 61131-3.

La figure 3 représente un exemple de déclaration de bloc fonction, par texte structuré (ST) et diagramme de bloc fonction (FBD).

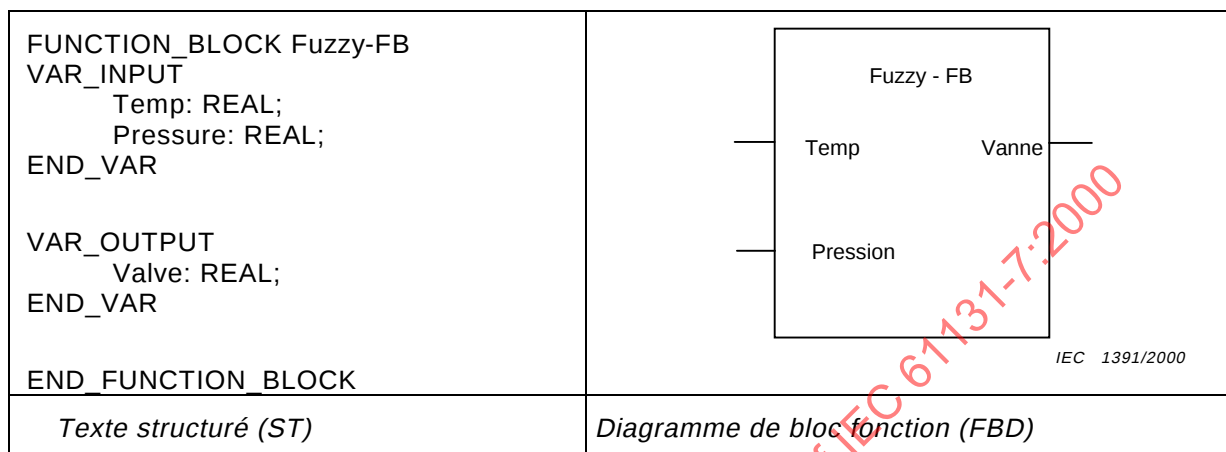


Figure 3 – Déclaration d'interface de bloc fonction en langage ST et FBD

5.2.2 Fuzzification

Les valeurs des variables d'entrée doivent être converties en *degrés d'appartenance* pour les *fonctions d'appartenance* définies sur la variable. Cette conversion est décrite entre les mots-clés FUZZIFY et END_FUZZIFY.

```

FUZZIFY variable_name
    TERM term_name:= membership_function;
    ....
END_FUZZIFY
        
```

Le mot-clé FUZZIFY est suivi du nom de la variable qui intervient dans la fuzzification. Ce nom est celui d'une variable précédemment définie dans la section VAR_INPUT. Cette *variable linguistique* doit être décrite par un ou plusieurs *termes linguistiques*. Les *termes linguistiques* sont introduits par le mot-clé TERM décrit par des *fonctions d'appartenance* en vue de fuzzifier la variable. Une *fonction d'appartenance* est une fonction linéaire par morceaux. Elle est définie par une table de points.

```

membership_function ::= (point i), (point j), ...
        
```

Chaque point correspond à une paire des valeurs de variable et degré d'appartenance de cette valeur, séparées par une virgule. Les paires sont entre parenthèses et séparées par des virgules.

```

point i ::= valeur d'entrée i | variable_name d'entrée i, valeur i de degré d'appartenance
        
```

Avec cette définition, tous les éléments simples, rampe et triangle par exemple, peuvent être définis. Les points doivent être indiqués par ordre croissant de la valeur de la variable. La fonction d'appartenance est linéaire entre deux points successifs. Le degré d'appartenance de chaque terme est donc calculé à partir de la valeur d'entrée «nette», par interpolation linéaire entre les deux points adjacents concernés de la fonction d'appartenance.

Un minimum de deux points est nécessaire. Le nombre maximal est limité suivant les classes de conformité de l'article 6.

With these language elements, it is possible to describe a function block interface. The function block interface is defined with parameters which are passed into and out of the function block. The data types of these parameters shall be defined according to IEC 61131-3.

Figure 3 shows an example of a Function Block declaration in Structured Text (ST) and Function Block Diagram (FBD) languages.

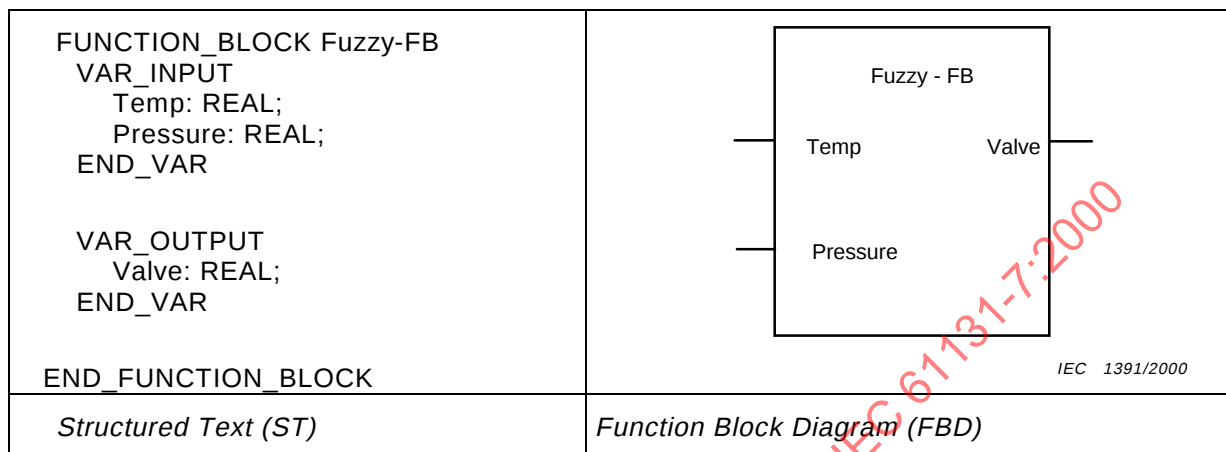


Figure 3 – Example of a Function Block interface declaration in ST and FBD languages

5.2.2 Fuzzification

The values of the input variables have to be converted into *degrees of membership* for the *membership functions* defined on the variable. This conversion is described between the keywords FUZZIFY and END_FUZZIFY.

```

FUZZIFY variable_name
  TERM term_name:= membership_function;
  ....
END_FUZZIFY
        
```

After the keyword FUZZIFY, the name of a variable which is used for the fuzzification shall be named. This is the name of a previously defined variable in the VAR_INPUT section. This *linguistic variable* shall be described by one or more *linguistic terms*. The *linguistic terms* introduced by the keyword TERM described by *membership functions* in order to fuzzify the variable. A *membership function* is a piece-wise linear function. It is defined by a table of points.

```
membership_function ::= (point i), (point j), ...
```

Every point is a pair of the values of the variable and the membership degree of that value separated by a comma. The pairs are enclosed in parentheses and separated by commas.

```
point i ::= value of input i / variable_name of input i , value i of membership degree
```

With this definition, all simple elements, for example ramp and triangle, may be defined. The points shall be given in ascending order of variable value. The membership function is linear between successive points. The degree of membership for each term is therefore calculated from the crisp input value by the linear interpolation between the two relevant adjacent membership function points.

The minimum number of points is two. The maximum number is restricted according to clause 6 conformance classes.

Exemple de *fonction d'appartenance* composée de trois points, assignée au *terme linguistique* «chaud»:

```
TERM warm:= (17.5, 0.0) (20.0, 1.0) (22.5, 0.0);
```

Si la valeur d'une *variable linguistique* est inférieure au premier point du tableau de calcul, toutes les valeurs inférieures au premier point du tableau doivent avoir le degré d'appartenance défini au premier point.

Si la valeur d'une *variable linguistique* est supérieure au dernier point de base du tableau de calcul, toutes les valeurs supérieures au dernier point du tableau doivent avoir le degré d'appartenance défini au dernier point.

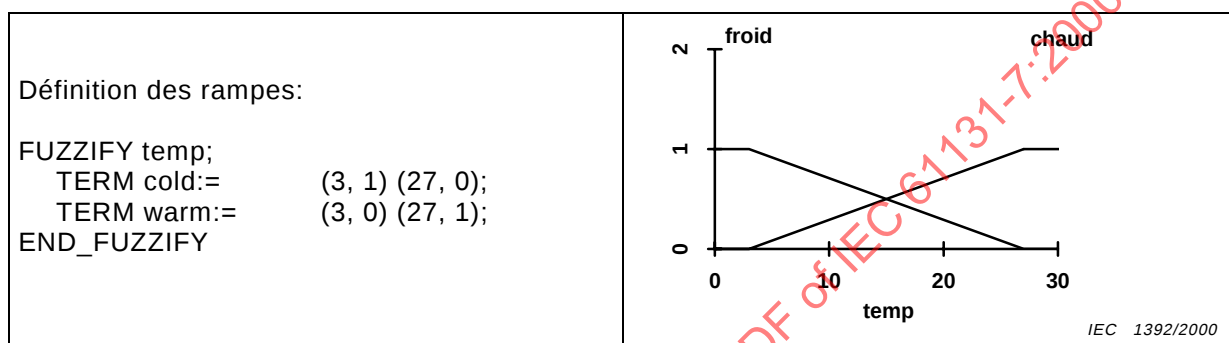


Figure 4 – Exemple de termes de rampe

NOTE Le type de données des points de fonctions d'appartenance n'est pas défini. Il faut que le fabricant fournisse un compilateur capable de prendre en charge toute conversion nécessaire.

Pour adapter l'application de contrôle flou en ligne, il est possible de modifier les points de base des fonctions d'appartenance. Cela peut être fait en utilisant les variables d'entrée du bloc fonction. Ces variables doivent être déclarées dans la section VAR_INPUT du bloc fonction. Un exemple d'utilisation de variables pour la définition des points de fonctions d'appartenance est illustré à la figure 5.

NOTE Certains points de fonction d'appartenance peuvent avoir des valeurs hors-séquence pendant l'exécution.

```

VAR_INPUT
  temp: REAL;          (* cette entrée doit être fuzzifiée *)
  pressure: REAL;      (* cette entrée doit être fuzzifiée *)
  bp_warm1, bp_warm2: REAL; (* entrées pour adaptation en ligne *)
END_VAR
FUZZIFY temp
  TERM warm:= (bp_warm1, 0.0), (21.0, 1.0), (bp_warm2, 0.0);
..
END_FUZZIFY

```

IEC 1393/2000

Figure 5 – Exemple d'utilisation de variables pour les fonctions d'appartenance

5.2.3 Défuzzification

La *variable linguistique* d'une variable de sortie doit être convertie en valeur numérique. Cette conversion est décrite entre les mots-clés DEFUZZIFY et END_DEFUZZIFY.

Le mot-clé DEFUZZIFY est suivi par la variable qui intervient dans la défuzzification. Ce nom est celui d'une variable précédemment définie dans la section VAR_OUTPUT.

Example of *membership function* with three points for *linguistic term* "warm":

```
TERM warm:= (17.5, 0.0) (20.0, 1.0) (22.5, 0.0);
```

If the value of a *linguistic variable* is less than the first base point in the look-up table, all values below the first point in the look-up table shall have the same membership degree as defined at the first point.

If the value of a *linguistic variable* is greater than the last base point in the look-up table, all values greater than the last point in the look-up table shall have the same membership degree as defined at the last point.

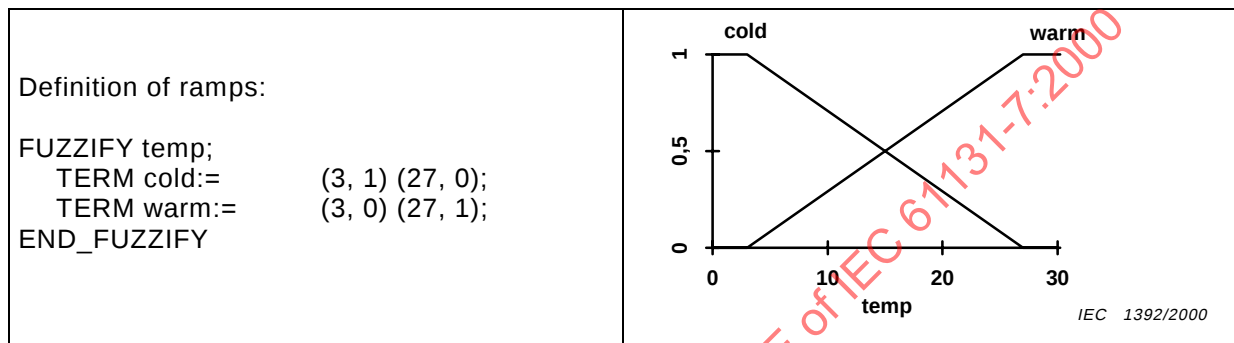


Figure 4 – Example of ramp terms

NOTE The data type of the points of membership functions is not defined. The manufacturer must provide a compiler that accommodates any necessary conversion.

In order to adapt the fuzzy control application on-line, the base points in the membership functions can be modified. This may be done using variables which are input to the function block. These variables have to be declared in the VAR_INPUT section of the function block. An example for the use of variables for the definition of the points for the membership functions is given in figure 5.

NOTE The values of membership function points at runtime may be out of sequence.

<pre> VAR_INPUT temp: REAL; (* this input shall be fuzzified *) pressure: REAL; (* this input shall be fuzzified *) bp_warm1, bp_warm2: REAL; (* these inputs are for on-line adaptation *) END_VAR FUZZIFY temp TERM warm:= (bp_warm1, 0.0), (21.0, 1.0), (bp_warm2, 0.0); .. END_FUZZIFY </pre>	
IEC 1393/2000	

Figure 5 – Example of usage of variables for membership functions

5.2.3 Defuzzification

A *linguistic variable* for an output variable has to be converted into a value. This conversion is described between the keywords DEFUZZIFY and END_DEFUZZIFY.

After the keyword DEFUZZIFY, the variable which is used for the *defuzzification* shall be named. This is the name of a previous defined variable in the VAR_OUTPUT section.

```

DEFUZZIFY variable_name
    RANGE(min..max);
    TERM term_name:= membership_function;
    defuzzification_method;
    default_value;
END_DEFUZZIFY
    
```

La définition des *termes linguistiques* est donnée en 5.2.2.

Les *singletons* sont des *fonctions d'appartenance* spéciales utilisées en sortie pour simplifier la *défuzzification*. Les singletons ne sont décrits que par une seule valeur pour le *terme linguistique*. Quelques exemples de termes sont présentés à la figure 6.

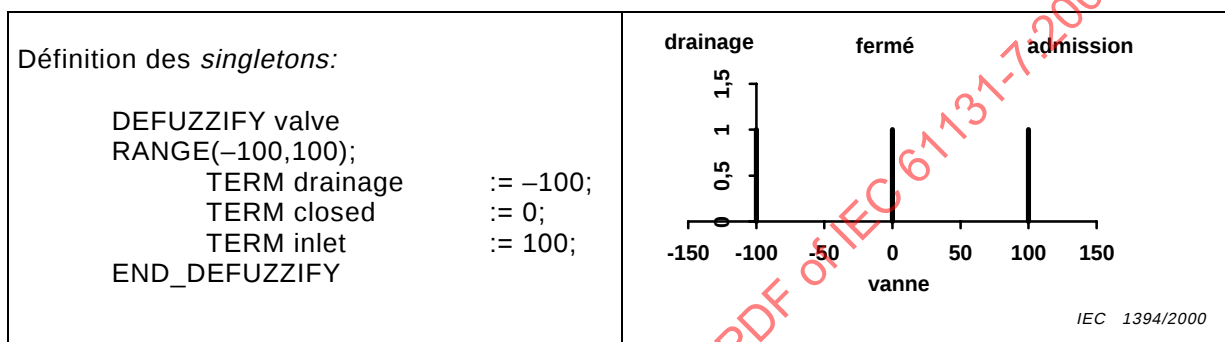


Figure 6 – Exemple de termes singletons

La méthode de défuzzification doit être définie par le mot clé METHOD.

```

METHOD: defuzzification_method;
    
```

Les méthodes de défuzzification suivantes sont possibles (voir tableaux 1 et 2).

Tableau 1 – Méthodes de défuzzification

Mot-clé	Description
CoG	Centre de gravité (note 1)
CoGS	Centre de gravité de singletons
CoA	Centre de surface (notes 2 et 3)
LM	Maximum le plus à gauche (note 4)
RM	Maximum le plus à droite (note 4)
NOTE 1 Centre de gravité équivalent à Centroïde de surface. NOTE 2 Centre de surface équivalent à Bissectrice de surface. NOTE 3 CoA non applicable en cas de singletons. NOTE 4 Les méthodes de défuzzification LM et RM sont asymétriques par rapport à zéro.	

```

DEFUZZIFY variable_name
    RANGE(min..max);
    TERM term_name:= membership_function;
    defuzzification_method;
    default_value;
END_DEFUZZIFY

```

The definition of *linguistic terms* is given in 5.2.2.

Singletons are special *membership functions* used for outputs in order to simplify the *defuzzification*. They are described only by a single value for the *linguistic term*. In figure 6, examples of terms are given.

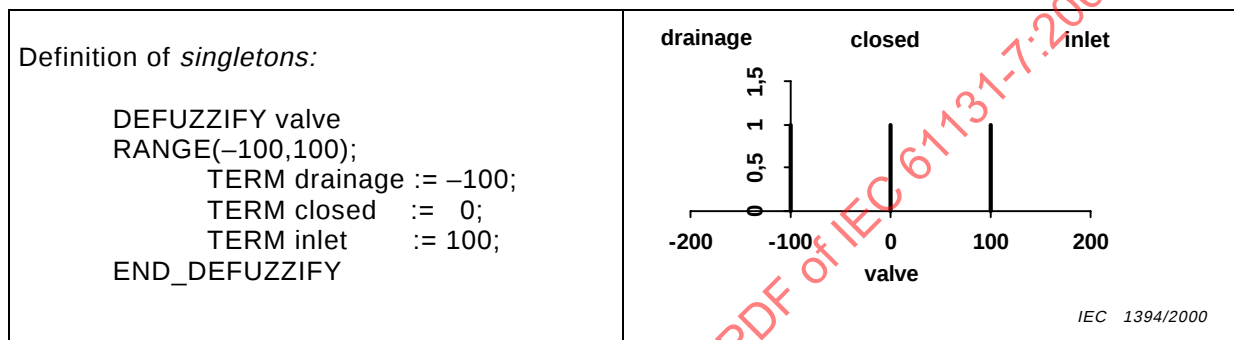


Figure 6 – Example of singleton terms

The defuzzification method shall be defined by the language element METHOD.

```

METHOD: defuzzification_method;

```

The following defuzzification methods are allowed (see tables 1 and 2).

Table 1 – Defuzzification methods

Keyword	Explanation
CoG	Centre of Gravity (note 1)
CoGS	Centre of Gravity for Singletons
CoA	Centre of Area (notes 2 and 3)
LM	Left Most Maximum (note 4)
RM	Right Most Maximum (note 4)
NOTE 1 Centre of Gravity is equivalent to Centroid of Area.	
NOTE 2 Centre of Area is equivalent to Bisector of Area.	
NOTE 3 CoA is not applicable if singletons are used.	
NOTE 4 LM and RM defuzzification methods are asymmetrical about zero.	

Tableau 2 – Formules des différentes méthodes de défuzzification

COG	$U = \frac{\int_{\text{Min}}^{\text{Max}} u \mu(u) du}{\int_{\text{Min}}^{\text{Max}} \mu(u) du}$
COGS	$U = \frac{\sum_{i=1}^p [u_i \mu_i]}{\sum_{i=1}^p [\mu_i]}$
COA	$U = u', \int_{\text{Min}}^{u'} \mu(u) du = \int_{u'}^{\text{Max}} \mu(u) du$
RM	$U = \sup (u'), \mu(u') = \sup_{u \in [\text{Min}, \text{Max}]} \mu(u)$
LM	$U = \inf (u'), \mu(u') = \inf_{u \in [\text{Min}, \text{Max}]} \mu(u)$

où:
 U est le résultat de défuzzification;
 u est la variable de sortie;
 p est le nombre de singletons;
 μ est la fonction d'appartenance des ensembles flous accumulés;
 i est l'index;
 Min est la valeur minimale de défuzzification définie par RANGE
 Dans le cas de singletons, Min = $-\infty$ (moins l'infini);
 Max est la valeur maximale de défuzzification définie par RANGE.
 Dans le cas de singletons, Max = $+\infty$ (plus l'infini);
 sup est la plus grande valeur;
 inf est la plus petite valeur.

Si le degré d'appartenance est égal à 0 pour tous les *termes linguistiques* d'une variable de sortie, cela signifie qu'aucune règle n'est active pour cette variable. Dans un tel cas, la défuzzification est incapable de produire une sortie valide; une valeur par défaut pour la sortie peut donc être définie. Cette valeur par défaut ne correspond à la valeur de variable de sortie que dans le cas où aucune règle n'est activée.

DEFAULT:= *value* | NC;

Le mot-clé DEFAULT doit être suivi par la valeur par défaut souhaitée. Sinon, le mot-clé NC (aucun changement) doit être ajouté pour indiquer que la sortie doit rester inchangée lorsqu'aucune règle n'est activée.

RANGE (intervalle) permet de spécifier une valeur minimale et une valeur maximale, séparées par deux points.

RANGE:= (*valeur minimum*..*valeur maximum*);

L'intervalle RANGE permet de limiter chaque fonction d'appartenance à la plage de valeurs de chaque variable de sortie. Si des singletons sont utilisés pour les fonctions d'appartenance de sortie, la valeur RANGE est sans effet.

Si aucun intervalle de valeurs n'est défini, un intervalle par défaut est utilisé, qui correspond au type de données de la variable spécifiée dans la CEI 61131-3.

Table 2 – Formulae for defuzzification methods

COG	$U = \frac{\int_{\text{Min}}^{\text{Max}} u \mu(u) du}{\int_{\text{Min}}^{\text{Max}} \mu(u) du}$
COGS	$U = \frac{\sum_{i=1}^p [u_i \mu_i]}{\sum_{i=1}^p [\mu_i]}$
COA	$U = u', \int_{\text{Min}}^{u'} \mu(u) du = \int_{u'}^{\text{Max}} \mu(u) du$
RM	$U = \sup (u'), \mu(u') = \sup_{u \in [\text{Min}, \text{Max}]} \mu(u)$
LM	$U = \inf (u'), \mu(u') = \inf_{u \in [\text{Min}, \text{Max}]} \mu(u)$

where

U is the result of defuzzification;

u is the output variable;

p is the number of singletons;

μ is the membership function of accumulated fuzzy sets;

i is the index;

Min is the minimum value for defuzzification defined in RANGE.
In the case of singletons, Min = –infinity;

Max is the maximum value for defuzzification defined in RANGE.
In the case of singletons, Max = +infinity;

sup is the largest value;

inf is the smallest value.

If the degree of membership is 0 for all *linguistic terms* of an output variable, that means: no rule for this variable is active. In that case, the *defuzzification* is not able to generate a valid output. Therefore, it is allowed to define a default value for the output. This default value is the value for the output variable only in the case when no rule has fired.

DEFAULT:= *value* | NC;

After the keyword DEFAULT, the value shall be specified. Otherwise, the keyword NC (no change) shall be specified to indicate that the output shall remain unchanged if no rule has fired.

The range is a specification of a minimum value and a maximum value separated by two points.

RANGE:= (minimum *value* .. maximum *value*);

The RANGE is used for limiting each membership function to the range of each output variable. If singletons are used for output membership functions, the RANGE has no effect.

If there is no range defined the default range shall be the range of the data type of the variable specified in IEC 61131-3.

5.2.4 Bloc de règles

L'inférence de l'algorithme flou doit être définie dans un ou plusieurs blocs de règle. Pour faciliter les manipulations et prévoir la possibilité de diviser la base de règles en différents modules, il est admis d'utiliser plusieurs blocs de règles. Chaque bloc de règles doit avoir un nom différent.

Les règles doivent être définies entre les mots-clés RULEBLOCK et END_RULEBLOCK.

```
RULEBLOCK ruleblock_name
    operator_definition;
    [activation_method;]
    accumulation_method;
    rules;
END_RULEBLOCK
```

Les opérateurs flous sont utilisés à l'intérieur du bloc de règles.

```
operator_definition ::= operator: algorithm
```

Pour satisfaire à la Loi de Morgan, les algorithmes correspondant aux opérateurs ET et OU doivent être utilisés par paires; par exemple MAX correspond à OU si MIN représente ET.

Tableau 3 – Algorithmes appariés

opérateur OU		opérateur ET	
mot-clé pour Algorithme	Algorithme	mot-clé pour Algorithme	Algorithme
MAX	$\text{Max}(\mu_1(x), \mu_2(x))$	MIN	$\text{Min}(\mu_1(x), \mu_2(x))$
ASUM	$\mu_1(x) + \mu_2(x) - \mu_1(x) \mu_2(x)$	PROD	$\mu_1(x) \mu_2(x)$
BSUM	$\text{Min}(1, \mu_1(x) + \mu_2(x))$	BDIF	$\text{Max}(0, \mu_1(x) + \mu_2(x) - 1)$

Exemple de blocs de règles:

```
RULEBLOCK first
    AND: MIN;
    ..
END_RULEBLOCK
RULEBLOCK second
    AND: PROD;
    ..
END_RULEBLOCK
```

La méthode d'activation est définie par l'élément de langage suivant:

```
ACT: activation_method;
```

Les méthodes d'activation suivantes sont possibles (voir tableau 4):

Tableau 4 – Méthodes d'activation

Nom	Mot-clé	Algorithme
Produit	PROD	$\mu_1(x) \mu_2(x)$
Minimum	MIN	$\text{Min}(\mu_1(x), \mu_2(x))$
NOTE La méthode d'activation ne s'applique pas aux singletons.		

5.2.4 Rule block

The inference of the fuzzy algorithm shall be defined in one or more rule blocks. For proper handling and to cater for the possibility of splitting the rule base into different modules, the use of several rule blocks is allowed. Each rule block shall have a unique name.

Rules shall be defined between the keywords RULEBLOCK and END_RULEBLOCK.

```
RULEBLOCK ruleblock_name
    operator_definition;
    [activation_method;]
    accumulation_method;
    rules;
END_RULEBLOCK
```

The fuzzy operators are used inside the rule block.

```
operator_definition ::= operator: algorithm
```

To fulfill de Morgan's Law, the algorithms for operators AND and OR shall be used pair-wise; for example MAX shall be used for OR if MIN is used for AND.

Table 3 – Paired algorithms

operator OR		operator AND	
keyword for Algorithm	Algorithm	keyword for Algorithm	Algorithm
MAX	$\text{Max}(\mu_1(x), \mu_2(x))$	MIN	$\text{Min}(\mu_1(x), \mu_2(x))$
ASUM	$\mu_1(x) + \mu_2(x) - \mu_1(x) \mu_2(x)$	PROD	$\mu_1(x) \mu_2(x)$
BSUM	$\text{Min}(1, \mu_1(x) + \mu_2(x))$	BDIF	$\text{Max}(0, \mu_1(x) + \mu_2(x) - 1)$

An example of rule blocks:

```
RULEBLOCK first
    AND: MIN;
    ..
END_RULEBLOCK
RULEBLOCK second
    AND: PROD;
    ..
END_RULEBLOCK
```

The following language element defines the method of the activation:

```
ACT: activation_method;
```

The following activation methods are may be used (see table 4):

Table 4 – Activation methods

Name	Keyword	Algorithm
Product	PROD	$\mu_1(x) \mu_2(x)$
Minimum	MIN	$\text{Min}(\mu_1(x), \mu_2(x))$
NOTE The activation method is not relevant for singletons.		

La méthode d'accumulation est définie par l'élément de langage suivant:

ACCU: <i>accumulation_method</i> ;

Les méthodes d'accumulation suivantes sont admises (voir tableau 5):

Tableau 5 – Méthodes d'accumulation

Nom	Mot-clé	Formule
Maximum	MAX	$\text{MAX } (\mu_1(x), \mu_2(x))$
Somme limitée	BSUM	$\text{MIN } (1, \mu_1(x) + \mu_2(x))$
Somme normalisée	NSUM	$\frac{\mu_1(x) + \mu_2(x)}{\text{MAX } (1, \text{MAX}_{x' \in X} (\mu_1(x') + \mu_2(x')))}$

Les entrées d'un bloc de règles sont des *variables linguistiques* associées à un ensemble de *termes linguistiques*. Un degré d'appartenance est assigné à chaque terme.

Les règles sont définies à l'intérieur du bloc de règles. Chacune commence par le mot-clé RULE (règle) suivi par le nom de la règle et doit se terminer par un point-virgule. Un numéro unique correspond à chaque règle à l'intérieur du bloc de règles.

RULE <i>numbers</i> : IF condition THEN conclusion [WITH facteur de pondération];

La règle elle-même doit débiter par le mot-clé IF (si) suivi par la *condition*. La *conclusion* est ensuite énoncée, débutant par le mot-clé THEN (alors).

Il est permis de combiner plusieurs *sous-conditions* et variables d'entrée dans une seule règle. Les variables permettent l'importation de degrés d'appartenance flous dans le bloc fonction flou. Toutes ces variables devront être définies entre les mots-clés IF et THEN, puis combinées par les opérateurs ET, OU ou PAS.

La priorité de l'opérateur (voir tableau 6) est traitée suivant l'algèbre booléenne décrite au tableau 3.

Tableau 6 – Priorité des opérateurs

Priorité	Opérateur
1	() parenthèses
2	PAS
3	ET
4	OU

Exemple de règle simplifié:

RULE 1: IF <i>subcondition1</i> AND <i>variable1</i> OR <i>variable2</i> THEN <i>conclusion</i> ;

Au niveau de conformité de base, l'opération OR peut être implémentée en définissant deux règles:

RULE 3: IF <i>subcondition 1</i> OR <i>subcondition 2</i> THEN <i>conclusion</i>; remplacée par: RULE 3a: IF <i>condition 1</i> THEN <i>conclusion</i>; RULE 3b: IF <i>condition 2</i> THEN <i>conclusion</i>;
--

The following language element defines the method of the accumulation:

ACCU: *accumulation_method*;

The following accumulation methods may be used (see table 5):

Table 5 – Accumulation methods

Name	Keyword	Formula
Maximum	MAX	$\text{MAX } (\mu_1(x), \mu_2(x))$
Bounded sum	BSUM	$\text{MIN } (1, \mu_1(x) + \mu_2(x))$
Normalized sum	NSUM	$\frac{\mu_1(x) + \mu_2(x)}{\text{MAX } (1, \text{MAX}_{x' \in X} (\mu_1(x') + \mu_2(x')))}$

The inputs of a rule block are *linguistic variables* with a set of *linguistic terms*. Each term has a degree of membership assigned to it.

The rules are defined inside the rule block. Each begins with the keyword RULE followed by a name for the rule and shall be concluded by a semicolon. Each rule has a unique number inside the rule block.

RULE *numbers*: IF condition THEN conclusion [WITH weighting factor];

The rule itself shall begin with the keyword IF followed by the *condition*. After the *condition*, the *conclusion* follows, beginning with the keyword THEN.

It is allowed to combine several *subconditions* and input variables in one rule. The purpose of variables is to permit fuzzy degrees of membership to be imported into the fuzzy Function Block. All of them shall be defined between the keywords IF and THEN, and combined by the operators with the keywords AND, OR or NOT.

The priority of the operator (see table 6) is handled according to Boolean algebra given in table 3.

Table 6 – Priority of operators

Priority	operator
1	() parenthesis
2	NOT
3	AND
4	OR

Simplified example for a rule:

RULE 1: IF *subcondition1* AND *variable1* OR *variable2* THEN *conclusion*;

In the Basic Level of conformance, the OR operation may be implemented by defining two rules:

RULE 3: IF *subcondition 1* OR *subcondition 2* THEN *conclusion*;
 replaced by:
 RULE 3a: IF *condition 1* THEN *conclusion*;
 RULE 3b: IF *condition 2* THEN *conclusion*;

La sous-condition commence par le nom d'une *variable linguistique*, suivi du mot-clé IS (est) et d'un NOT (pas) facultatif, et comprend un *terme linguistique* de la *variable linguistique* utilisée dans la *condition*.

Subcondition:= variable_linguistique IS [NOT] terme_linguistique

Les *termes linguistiques* utilisés dans la *condition* doivent correspondre à la *variable linguistique* pour la même *condition*. Le terme utilisé doit d'abord être défini par le mot-clé TERM.

Exemple de sous-conditions:

temp IS hot (la température EST élevée)
temp IS NOT hot (la température N'EST PAS élevée)

Il est également possible d'utiliser le mot-clé NOT (pas) devant la sous-condition. Dans ce cas, des parenthèses peuvent être utilisées.

IF NOT temp IS hot THEN ... ou IF NOT (temp IS hot) THEN ...

La *conclusion* peut être décomposée en plusieurs sous-conclusions et variables de sortie.

La sous-conclusion commence par le nom d'une *variable linguistique* suivi du mot-clé IS (est) et d'un *terme linguistique* provenant de la *variable linguistique* donnée.

Subconclusion:= variable_linguistique IS terme_linguistique

Exemple contenant plusieurs sous-conclusions sur une ou plusieurs lignes:

IF temp IS cold AND pressure IS low THEN var1, valve1 IS inlet , valve2 IS closed;
ou sur plusieurs lignes:
IF temp IS cold AND pressure IS low
THEN var1,
valve1 IS inlet,
valve2 IS closed;

En option, il est admis d'assigner un *facteur de pondération* à chaque sous-conclusion, qui est soit une valeur comprise entre 0.0 et 1.0, soit une variable. Le mot-clé WITH (avec) est ajouté, suivi du *facteur de pondération*.

Le rôle du *facteur de pondération* est de réduire le degré d'appartenance (fonction d'appartenance) de la sous-conclusion, par multiplication du résultat de la sous-conclusion par le *facteur de pondération*.

Pour manipuler les paramètres d'application de contrôle flou de l'extérieur, le *facteur de pondération* peut être représenté par une variable. Dans ce cas, la variable doit être déclarée dans la section VAR_INPUT. Cette option permet de modifier le *facteur de pondération* pendant l'exécution en vue d'adapter le programme de contrôle flou aux besoins du processus.

Si aucun énoncé WITH (avec) n'est assigné à la sous-conclusion, l'hypothèse d'un *facteur de pondération* par défaut de 1.0 doit être utilisée.

IF condition THEN subconclusion [WITH weighting_factor] subconclusion;

Exemple de *facteur de pondération* constant:

IF temp IS cold AND pressure IS low THEN valve1 IS inlet WITH 0.5,
valve2 IS closed;

The subcondition begins with the name of a *linguistic variable* followed by the keyword IS with an optional NOT and one *linguistic term* of the *linguistic variable* used in the *condition*.

Subcondition := *linguistic_variable* IS [NOT] *linguistic_term*

The *linguistic terms* which are used in the *condition* shall match the *linguistic variable* in the same *condition*. The term used has to be previously defined with the keyword TERM.

Example of subconditions:

temp IS hot
temp IS NOT hot

It is also possible to use the keyword NOT in front of the subcondition. In this case, parentheses may be used.

IF NOT temp IS hot THEN ... or IF NOT (temp IS hot) THEN ...

The *conclusion* may be split into several subconclusions and output variables.

The subconclusion begins with the name of a *linguistic variable* followed by the keyword IS and one *linguistic term* of the given *linguistic variable*.

Subconclusion := *linguistic_variable* IS *linguistic_term*

Example with several subconclusions in one or more lines:

IF temp IS cold AND pressure IS low THEN var1, valve1 IS inlet , valve2 IS closed;
or in several lines:
IF temp IS cold AND pressure IS low
THEN var1,
valve1 IS inlet,
valve2 IS closed;

Optionally it is allowed to give each subconclusion a *weighting factor* which is a number with a value between 0.0 and 1.0 or a variable. This shall be done by the keyword WITH followed by the *weighting factor*.

The *weighting factor* shall reduce the membership degree (membership function) of the subconclusion by multiplication of the result in the subconclusion with the *weighting factor*.

In order to manipulate the fuzzy control application parameters externally, the *weighting factor* may be a variable. In this case, the variable has to be declared in the VAR_INPUT section. This enables the possibility to change the *weighting factor* during runtime in order to adapt the fuzzy control program to process needs.

If there is no WITH statement assigned to the subconclusion, a default *weighting factor* of 1.0 shall be assumed.

IF *condition* THEN *subconclusion* [WITH *weighting_factor*] *subconclusion*;

An example of a constant *weighting_factor*:

IF temp IS cold AND pressure IS low THEN valve1 IS inlet WITH 0.5,
valve2 IS closed;

Exemple de *facteur de pondération* variable:

```
VAR_INPUT
    w_myrule1: REAL:= 0.8;
END_VAR
RULEBLOCK temp_rule
    RULE 1:      IF temp      IS cold AND pressure IS low
                  THEN valve  IS inlet WITH w_myrule1;
    ..
END_RULEBLOCK
```

5.2.5 Paramètres facultatifs

L'implémentation sur différents systèmes cibles peut nécessiter des informations supplémentaires à fournir au système afin de permettre la meilleure conversion possible des applications de contrôle flou.

De telles informations supplémentaires peuvent être nécessaires pour un élément de langage placé entre les mots-clés OPTIONS et END_OPTIONS.

```
OPTIONS
    application_specific_parameters
END_OPTIONS
```

Ces éléments de langage doivent intervenir pour des caractéristiques correspondant à la classe de conformité de niveau ouvert, selon l'article 6.

IECNORM.COM : Click to view the full PDF of IEC 61131-7:2000

An example of a variable *weighting factor*:

```
VAR_INPUT
    w_myrule1: REAL:= 0.8;
END_VAR
RULEBLOCK temp_rule
    RULE 1:      IF temp      IS cold AND pressure IS low
                  THEN valve  IS inlet WITH w_myrule1;
    ..
END_RULEBLOCK
```

5.2.5 Optional parameters

For implementation on different target systems, it may be necessary to give additional information to the system in order to allow the best possible conversion of fuzzy control applications.

Such additional information may be required in a language element enclosed by OPTIONS and END_OPTIONS.

```
OPTIONS
    application_specific_parameters
END_OPTIONS
```

These language elements shall be used for features in the conformance class of the open level according to clause 6.

IECNORM.COM : Click to view the full PDF of IEC 61131-7:2000

5.3 Exemple de FCL

Un exemple de langage de contrôle flou est présenté à la figure 7.

```

FUNCTION_BLOCK Fuzzy_FB
VAR_INPUT
    temp:    REAL;
    pressure: REAL;
END_VAR
VAR_OUTPUT
    valve:    REAL;
END_VAR
FUZZIFY temp
    TERM cold    := (3, 1) (27, 0);
    TERM hot     := (3, 0) (27, 1);
END_FUZZIFY
FUZZIFY pressure
    TERM low     := (55, 1) (95, 0);
    TERM high    := (55, 0) (95, 1);
END_FUZZIFY
DEFUZZIFY valve
    TERM drainage := -100;
    TERM closed   := 0;
    TERM inlet    := 100;
    METHOD: CoGS;
    DEFAULT       := 0;
END_DEFUZZIFY
RULEBLOCK No1
    AND: MIN;
    ACCU: MAX;
    RULE 1: IF temp IS cold AND pressure IS low      THEN valve IS inlet;
    RULE 2: IF temp IS cold AND pressure IS high     THEN valve IS closed
                                                    WITH 0.8;
    RULE 3: IF temp IS hot AND pressure IS low       THEN valve IS closed;
    RULE 4: IF temp IS hot AND pressure IS high      THEN valve IS drainage;
END_RULEBLOCK
END_FUNCTION_BLOCK
    
```

IEC 1395/2000

Figure 7 – Exemple de bloc fonction flou

5.4 Règles de production et mots-clés du langage de contrôle flou (FCL)

L'annexe A de la CEI 61131-3 définit la méthode de spécification de langage littéral pour les automates programmables. Cette méthode de spécification est appliquée ici pour le FCL.

L'annexe B de la CEI 61131-3 définit le cahier des charges formel des éléments de langage pour les langages de programmation littéraux de la CEI 61131-3. Pour le FCL, un sous-ensemble des éléments de langage suivants de l'annexe B de la CEI 61131-3 est utilisé:

- B.1.1 Lettres, chiffres et identificateurs
- B.1.2 Constantes
- B.1.3 Types de données
- B.1.4 Variables

5.3 FCL example

An example in Fuzzy Control Language is given in figure 7.

```

FUNCTION_BLOCK Fuzzy_FB
VAR_INPUT
    temp:    REAL;
    pressure: REAL;
END_VAR
VAR_OUTPUT
    valve:    REAL;
END_VAR
FUZZIFY temp
    TERM cold    := (3, 1) (27, 0);
    TERM hot     := (3, 0) (27, 1);
END_FUZZIFY
FUZZIFY pressure
    TERM low     := (55, 1) (95, 0);
    TERM high    := (55, 0) (95, 1);
END_FUZZIFY
DEFUZZIFY valve
    TERM drainage := -100;
    TERM closed   := 0;
    TERM inlet    := 100;
    METHOD: CoGS;
    DEFAULT       := 0;
END_DEFUZZIFY
RULEBLOCK No1
    AND: MIN;
    ACCU: MAX;
    RULE 1: IF temp IS cold AND pressure IS low THEN valve IS inlet;
    RULE 2: IF temp IS cold AND pressure IS high THEN valve IS closed WITH 0.8;
    RULE 3: IF temp IS hot AND pressure IS low THEN valve IS closed;
    RULE 4: IF temp IS hot AND pressure IS high THEN valve IS drainage;
END_RULEBLOCK
END_FUNCTION_BLOCK

```

IEC 1395/2000

Figure 7 – Example for fuzzy function block

5.4 Production rules and keywords of the Fuzzy Control Language (FCL)

Annex A of IEC 61131-3 defines the specification method for textual languages for programmable controllers. This specification method is used here for FCL.

Annex B of IEC 61131-3 defines the formal specification of language elements for the textual programming languages of IEC 61131-3. For FCL the following subset of language elements of annex B of IEC 61131-3 is used:

- B.1.1 Letters, digits and identifiers
- B.1.2 Constants
- B.1.3 Data types
- B.1.4 Variables

5.4.1 Règles de production

Les éléments de langage de la CEI 61131-3 évoqués plus haut peuvent être complétés par les éléments de langage suivants:

function_block_declaration ::= 'FUNCTION_BLOCK' function_block_name
 {fb_io_var_declarations}
 {other_var_declarations}
 function_block_body
 'END_FUNCTION_BLOCK'

fb_io_var_declarations ::= input_declarations | output_declarations

other_var_declarations ::= var_declarations

function_block_body ::= {fuzzify_block}
 {defuzzify_block}
 {rule_block}
 {option_block}

fuzzify_block ::= 'FUZZIFY' variable_name
 {linguistic_term}
 'END_FUZZIFY'

defuzzify_block ::= 'DEFUZZIFY' f_variable_name
 [range]
 {linguistic_term}
 defuzzification_method
 default_value
 'END_DEFUZZIFY'

rule_block ::= 'RULEBLOCK' nom_rule_block
 operator_definition
 [activation_method]
 accumulation_method
 {rule}
 'END_RULEBLOCK'

option_block ::= 'OPTION'
 paramètre spécifique au fabricant
 'END_OPTION'

linguistic_term ::= 'TERM' term_name ':= ' membership_function ','

membership_function ::= singleton | points

NOTE Se reporter à 5.2.3 pour une description des singletons.

singleton ::= numeric_literal | variable_name

points ::= {'('numeric_literal | variable_name ','
 numeric_literal ')')}

NOTE Se reporter à 5.2.2 pour connaître le nombre de points autorisé.

defuzzification_method ::= 'METHOD' ':' 'CoG' | 'CoGS' | 'CoA' | 'LM' | 'RM' ':'

default_value ::= 'DEFAULT' ':=' numeric_literal | 'NC' ':'

5.4.1 Production rules

Additionally to the above listed language elements of IEC 61131-3, the following language elements may be used:

```
function_block_declaration ::= 'FUNCTION_BLOCK' function_block_name
                             {fb_io_var_declarations}
                             {other_var_declarations}
                             function_block_body
                             'END_FUNCTION_BLOCK'
```

```
fb_io_var_declarations ::= input_declarations | output_declarations
```

```
other_var_declarations ::= var_declarations
```

```
function_block_body ::= {fuzzify_block}
                       {defuzzify_block}
                       {rule_block}
                       {option_block}
```

```
fuzzify_block ::= 'FUZZIFY' variable_name
                 {linguistic_term}
                 'END_FUZZIFY'
```

```
defuzzify_block ::= 'DEFUZZIFY' f_variable_name
                   [range]
                   {linguistic_term}
                   defuzzification_method
                   default_value
                   'END_DEFUZZIFY'
```

```
rule_block ::= 'RULEBLOCK' rule_block_name
              operator_definition
              [activation_method]
              accumulation_method
              {rule}
              'END_RULEBLOCK'
```

```
option_block ::= 'OPTION'
                 any manufacturer specific parameter
                 'END_OPTION'
```

```
linguistic_term ::= 'TERM' term_name ':=' membership_function ','
```

```
membership_function ::= singleton | points
```

NOTE For use of singletons, refer to 5.2.3.

```
singleton ::= numeric_literal | variable_name
```

```
points ::= {'(numeric_literal | variable_name ','
           numeric_literal ')}'
```

NOTE Refer to 5.2.2 for the allowed number of points.

```
defuzzification_method ::= 'METHOD' ':' 'CoG' | 'CoGS' | 'CoA' | 'LM' | 'RM' ':'
```

```
default_value ::= 'DEFAULT' ':=' numeric_literal | 'NC' ':'
```

range ::= 'RANGE('numeric_literal' '..' numeric_literal)' ';'

operator_definition ::= [('OR' ':' 'MAX' | 'ASUM' | 'BSUM')]
[('AND' ':' 'MIN' | 'PROD' | 'BDIF')] ';'

NOTE Les algorithmes appariés sont présentés au tableau 3.

activation_method ::= 'ACT' ':' 'PROD' | 'MIN' ';'

accumulation_method ::= 'ACCU' ':' 'MAX' | 'BSUM' | 'NSUM' ';'

rule ::= 'RULE' integer_literal ':'
'IF' condition 'THEN' conclusion [WITH weighting_factor] ';'

condition ::= x{ ('AND' x) | ('OR' x) }

x ::= ['NOT'] (subcondition | (' (' condition ') '))

subcondition ::= variable_name | (variable_name 'IS' ['NOT'] term_name)

conclusion ::= { (variable_name | (variable_name 'IS' term_name)) ',' }
(variable_name | variable_name 'IS' term_name)

weighting_factor ::= variable | numeric_literal

function_block_name ::= identifiant

ruleblock_name ::= identifiant

term_name ::= identifiant

variable_name_f ::= identifiant

variable_name ::= identifiant

numeric_literal ::= integer_literal | real_literal

input_declarations ::= voir CEI 61131-3, annexe B

output_declarations ::= voir CEI 61131-3, annexe B

var_declarations ::= voir CEI 61131-3, annexe B

identifiant ::= voir CEI 61131-3, annexe B

range ::= 'RANGE('numeric_literal' '..' numeric_literal)' ';'

operator_definition ::= [('OR' ':' 'MAX' | 'ASUM' | 'BSUM')]
[('AND' ':' 'MIN' | 'PROD' | 'BDIF')] ';'

NOTE See table 3 for paired algorithms.

activation_method ::= 'ACT' ':' 'PROD' | 'MIN' ';'

accumulation_method ::= 'ACCU' ':' 'MAX' | 'BSUM' | 'NSUM' ';'

rule ::= 'RULE' integer_literal ':'
'IF' condition 'THEN' conclusion [WITH weighting_factor] ';'

condition ::= x{('AND' x)}('OR' x) }

x ::= ['NOT'] (subcondition | (' (' condition ') '))

subcondition ::= variable_name | (variable_name 'IS' ['NOT'] term_name)

conclusion ::= { (variable_name | (variable_name 'IS' term_name)) ' , ' }
(variable_name | variable_name 'IS' term_name)

weighting_factor ::= variable | numeric_literal

function_block_name ::= identifier

ruleblock_name ::= identifier

term_name ::= identifier

f_variable_name ::= identifier

variable_name ::= identifier

numeric_literal ::= integer_literal | real_literal

input_declarations ::= see IEC 61131-3, annex B

output_declarations ::= see IEC 61131-3, annex B

var_declarations ::= see IEC 61131-3, annex B

identifier ::= see IEC 61131-3, annex B

5.4.2 Mots-clés

Tableau 7 – Mots-clés réservés pour le FCL

Mot-clé	Signification	Paragraphe
()	Parenthèses dans la condition, le terme, l'intervalle	5.2.4
ACCU	Méthode d'accumulation	5.2.4
ACT	Méthode d'activation	5.2.4
AND	Opérateur ET	5.2.4
ASUM	Opérateur OU, somme algébrique	5.2.4
BDIF	Opérateur ET, différence bornée	5.2.4
BSUM	Méthode d'accumulation, opérateur OU, somme bornée	5.2.4
CoA	Méthode de défuzzification par centre de surface	5.2.3
CoG	Méthode de défuzzification par centre de gravité	5.2.3
CoGS	Défuzzification de singletons par centre de gravité	5.2.3
DEFAULT	Valeur de sortie par défaut lorsqu'aucune règle n'est activée	5.2.3
DEFUZZIFY	Défuzzification de variable de sortie	5.2.3
END_DEFUZZIFY	Fin des spécifications de défuzzification	5.2.2
END_FUNCTION_BLOCK	Fin des spécifications de bloc fonction	5.2.1
END_FUZZIFY	Fin des spécifications de fuzzification	5.2.2
END_OPTIONS	Fin des spécifications d'options	5.2.5
END_RULEBLOCK	Fin des spécifications de bloc de règles	5.2.4
END_VAR	Fin des définitions de variable d'entrée/sortie	5.2.1
FUNCTION_BLOCK	Fin des spécifications de bloc fonction	5.2.1
FUZZIFY	Fuzzification de variable d'entrée	5.2.2
IF	Début de règle, suivi par la condition	5.2.4
IS	Suit la variable linguistique dans la condition / conclusion	5.2.4
LM	Méthode de défuzzification du maximum le plus à gauche	5.2.3
MAX	Méthode d'accumulation «maximum», opérateur OU	5.2.4
METHOD	Méthode de défuzzification	5.2.3
MIN	«Minimum», opérateur ET, méthode d'activation	5.2.4
NC	Aucune variation de variable de sortie en cas de non-activation d'une règle	5.2.3
NOT	Opérateur PAS	5.2.4
NSUM	Méthode d'accumulation par somme normalisée	5.2.4
OPTIONS	Définition de paramètres optionnels	5.2.5
OR	Opérateur OU	5.2.4
PROD	«Produit», opérateur ET, méthode d'activation	5.2.4
RANGE	Limites de fonctions d'appartenance	5.2.3
RM	Méthode de défuzzification du maximum le plus à droite	5.2.3
RULE	Début de spécification de règle floue	5.2.4
RULEBLOCK	Début de spécification de bloc de règles	5.2.4
TERM	Définition d'un terme linguistique (fonction d'appartenance) pour une variable linguistique	5.2.2
THEN	Sépare la condition de la conclusion	5.2.4
VAR	Définition de variable(s) locale(s)	5.2.1
VAR_INPUT	Définition de variable(s) d'entrée	5.2.1
VAR_OUTPUT	Définition de variable(s) de sortie	5.2.1
WITH	Définition du facteur de pondération	5.2.4

5.4.2 Keywords

Table 7 – Reserved keywords for FCL

Keyword	Meaning	Subclause
()	Parentheses in condition, term, range	5.2.4
ACCU	Accumulation method	5.2.4
ACT	Activation method	5.2.4
AND	AND operator	5.2.4
ASUM	OR operator, Algebraic sum	5.2.4
BDIF	AND operator, Bounded difference	5.2.4
BSUM	Accumulation method, OR operator, Bounded sum	5.2.4
CoA	Centre of area defuzzification method	5.2.3
CoG	Centre of gravity defuzzification method	5.2.3
CoGS	Centre of gravity defuzzification of singletons	5.2.3
DEFAULT	Default output value in case no rule has fired	5.2.3
DEFUZZIFY	Defuzzification of output variable	5.2.3
END_DEFUZZIFY	End of defuzzification specifications	5.2.3
END_FUNCTION_BLOCK	End of function block specifications	5.2.1
END_FUZZIFY	End of fuzzification specifications	5.2.2
END_OPTIONS	End of options specifications	5.2.5
END_RULEBLOCK	End of rule block specifications	5.2.4
END_VAR	End of input/output variable definitions	5.2.1
FUNCTION_BLOCK	End of function block specifications	5.2.1
FUZZIFY	Fuzzification of input variable	5.2.2
IF	Begin of rule which is followed by the condition	5.2.4
IS	Follows linguistic variable in condition and conclusion	5.2.4
LM	Left Most Maximum defuzzification method	5.2.3
MAX	Maximum accumulation method, OR operator	5.2.4
METHOD	Method of defuzzification	5.2.3
MIN	Minimum as AND operator, activation method	5.2.4
NC	No Change of output variable in case no rule has fired	5.2.3
NOT	NOT operator	5.2.4
NSUM	Normalised sum accumulation method	5.2.4
OPTIONS	Definition of optional parameters	5.2.5
OR	OR operator	5.2.4
PROD	Product as AND operator, activation method	5.2.4
RANGE	Limits for membership functions	5.2.3
RM	Right Most Maximum defuzzification method	5.2.3
RULE	Begin of specification of fuzzy rule	5.2.4
RULEBLOCK	Begin of specification of rule block	5.2.4
TERM	Definition of a linguistic term (membership function) for a linguistic variable	5.2.2
THEN	Separates condition from conclusion	5.2.4
VAR	Definition of local variable (s)	5.2.1
VAR_INPUT	Definition of input variable(s)	5.2.1
VAR_OUTPUT	Definition of output variable(s)	5.2.1
WITH	Definition of weighting factor	5.2.4

6 Conformité

6.1 Classes de conformité du langage de contrôle flou FCL

Les niveaux de conformité applicables aux systèmes de contrôle à langage de contrôle flou FCL sont indiqués à la figure 8. La hiérarchie se compose des trois niveaux suivants:

- Niveau Base, qui comprend les définitions de bloc fonction et types de données de la CEI 61131-3.
- Niveau Extension, dont les fonctionnalités optionnelles sont énumérées au tableau 9 et admises en option.
- Niveau Ouvert, qui englobe des fonctionnalités supplémentaires, non définies dans la présente partie de la CEI 61131, mais énumérées par le fabricant.

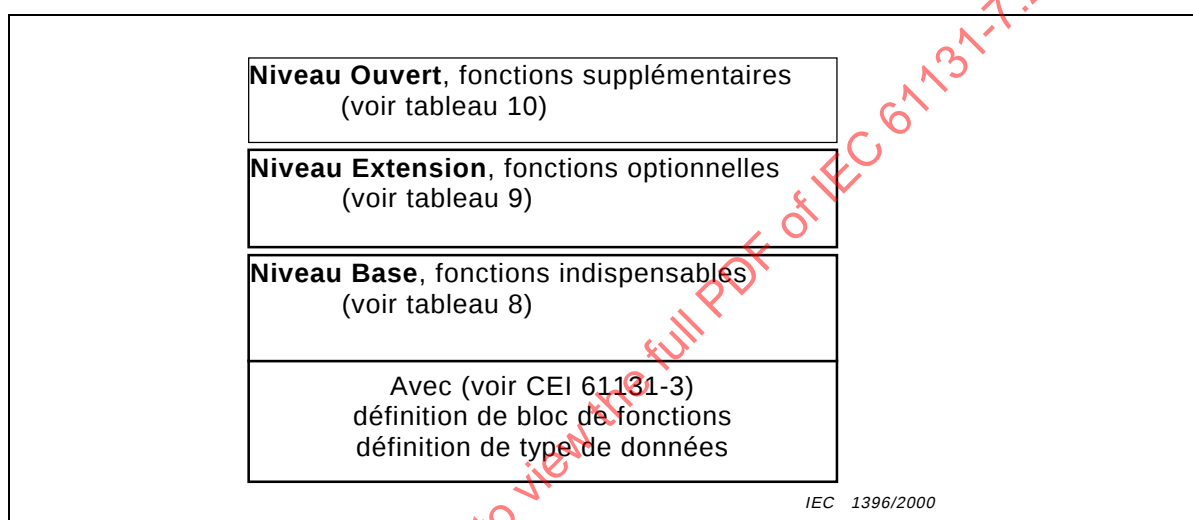


Figure 8 – Niveaux de conformité

Tout système de contrôle à langage de contrôle flou FCL qui revendique une conformité à la présente partie de la CEI 61131 doit satisfaire aux exigences suivantes:

- a) le système doit utiliser les caractéristiques de bloc fonction selon la CEI 61131-3 pour mettre en œuvre les fonctionnalités de contrôle flou. Par conséquent, les blocs fonction et les types de données nécessaires pour les paramètres d'entrée et sortie du bloc fonction doivent être définis conformément à la CEI 61131-3;
- b) toutes les caractéristiques de fonctionnalité de contrôle flou définies au tableau 8 doivent être implémentées selon les dispositions de la présente partie. Ce tableau définit l'ensemble des éléments de niveau de base communs à tous les systèmes conformes à la présente partie de la CEI 61131;
- c) un sous-ensemble d'éléments de niveau «extension» définis au tableau 9 contient des éléments supplémentaires qui peuvent être implémentés en option, en plus de toutes les caractéristiques de niveau de base. L'implémentation doit être strictement conforme aux dispositions de la présente partie. Ces caractéristiques doivent être marquées comme des extensions de norme sur une liste de caractéristiques mises en œuvre, sous la forme indiquée au tableau 9, incorporée à la documentation du système;
- d) d'autres caractéristiques au-delà des niveaux de base et d'extension peuvent être réalisées, dans la mesure où les fonctionnalités ou la représentation de ces caractéristiques diffèrent des caractéristiques standard, afin d'éviter toute confusion. Ces caractéristiques doivent être marquées comme des caractéristiques de niveau ouvert sur une liste sous la forme indiquée au tableau 10 qui doit faire partie de la documentation du système;

6 Compliance

6.1 Conformance classes of Fuzzy Control Language FCL

The levels of conformance for Control System using the Fuzzy Control Language (FCL) are shown in figure 8. The hierarchy consists of the following three levels.

- Basic Level including the definitions of the function block and the data types of IEC 61131-3.
- Extension level with optional features, listed in table 9 and optionally allowed within this level.
- Open Level encompassing additional features, not defined in this part of IEC 61131, but listed by the manufacturer.

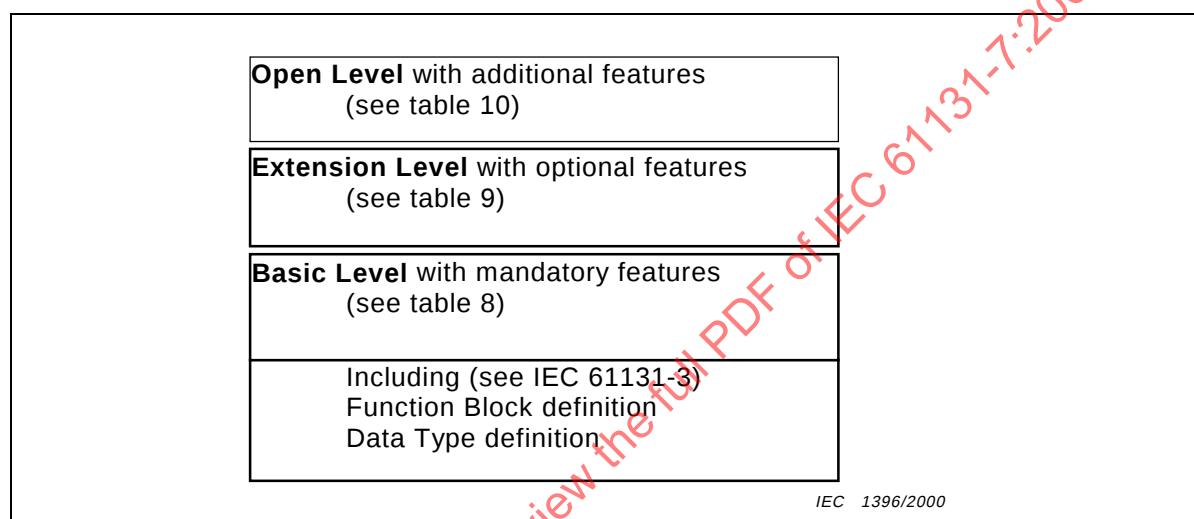


Figure 8 – Levels of conformance

A Control System using the Fuzzy Control Language FCL claiming to conform with this part of IEC 61131 shall comply with the following rules:

- a) it shall use the Function Block features according to IEC 61131-3 in order to realise the fuzzy control functionality. Therefore, the definition of the Function Blocks and the Data Types required for the input and output parameters of the fuzzy control function block shall be in accordance with IEC 61131-3;
- b) all features of fuzzy control functionality defined in table 8 shall be implemented according to the provisions of this part. This table defines the set of Basic Level elements which all control systems complying with this part of IEC 61131 shall have in common;
- c) a subset of the Extension Level elements defined in table 9 are additional elements which may be implemented optionally, in addition to all features of the Basic Level. The implementation shall strictly comply with the provisions of this part. These features shall be marked as Standard Extensions, and a list of realised features in the form of table 9 shall be part of the system documentation;
- d) further features exceeding the Basic Level and the Extended Level may be realized provided these features do not have the same or similar functionality or representation of the standard features, thereby avoiding any possible confusion. These features shall be marked as Open Level features, and a list in form of table 10 shall be part of the system documentation;

- e) l'échange de programmes d'application entre différents systèmes de contrôle flou doit être effectué dans la forme textuelle du langage de contrôle flou FCL, selon les dispositions de la présente norme. Cette présentation doit être disponible sur les systèmes conformes à la présente partie de la CEI 61131 comme présentation d'entrée et de sortie;
- f) afin de réaliser l'interface utilisateur la plus confortable et la mieux adaptée et ne pas gêner les progrès futurs, la représentation externe (conception, entrée, essais, etc.) d'un programme d'application de contrôle flou peut être réalisée par n'importe quel procédé graphique ou textuel.

Les éléments de langage du tableau 8 constituent un ensemble de base de fonctionnalités devant être intégrées à tout système de contrôle flou conforme à la présente partie de la CEI 61131.

Tableau 8 – Eléments de langage de contrôle flou FCL Niveau Base (obligatoires)

Elément de langage	Mot-clé	Détails
déclaration de bloc fonction	VAR_INPUT, VAR_OUTPUT	contient des variables d'entrée et sortie
fonction d'appartenance	variable d'entrée: TERM	maximum de trois points constants (coordonnée de degré d'appartenance = 0 ou 1)
	variable de sortie: TERM	singletons constants seulement
agrégation conditionnelle	opérateur: AND	algorithme: MIN
activation	–	sans objet car seuls des singletons sont utilisés
accumulation (agrégation de résultats)	opérateur: ACCU	algorithme: MAX
défuzzification	METHOD	algorithme: CoGS
valeur par défaut	DEFAULT	NC, valeur
bloc de règles	RULEBLOCK	un bloc de règles seulement
condition	IF ... IS ...	n sous-conditions
conclusion	THEN	une sous-conclusion seulement

Les éléments du tableau 9 constituent les caractéristiques d'extension, qui peuvent être mises en œuvre en option sur un système de contrôle flou standard (par exemple pour l'opérateur ET, on pourrait choisir l'algorithme PROD, BDIF ou les deux). Ces caractéristiques facultatives viennent compléter les caractéristiques de niveau de base.

- e) the exchange of application programs among different fuzzy control systems shall be done in the textual form of the Fuzzy Control Language FCL according to the provisions in this standard. This format shall be made available on systems complying with this part of IEC 61131 as input and output formats;
- f) In order to achieve the most comfortable and suitable user interface and to not hinder future progress, the external representations for the design, input, testing, etc. of fuzzy control application programs may be realized by any graphical or textual means.

The elements in table 8 are the basic set of features, which shall be realized in all fuzzy control systems complying with this part of IEC 61131.

Table 8 – FCL Basic Level language elements (mandatory)

Language element	Keyword	Details
function block declaration	VAR_INPUT, VAR_OUTPUT	contains input and output variables
membership function	input variable: TERM	maximum of three constant points (degree of membership coordinate = 0 or 1)
	output variable: TERM	constant singletons only
conditional aggregation	operator: AND	algorithm: MIN
activation	–	Not relevant because singletons are used only
accumulation (result aggregation)	operator: ACCU	algorithm: MAX
defuzzification	METHOD	algorithm: CoGS
default value	DEFAULT	NC value
ruleblock	RULEBLOCK	one ruleblock only
condition	IF ... IS ...	n subconditions
conclusion	THEN	only one subconclusion

The elements in table 9 are the extended set of features, which may be optionally realized in a standard fuzzy control system (e.g. for the AND operator the algorithm PROD or BDIF or both might be chosen). These optional features are in addition to all features of the Basic Level.

Tableau 9 – Éléments de langage de contrôle flou FCL Niveau Extension (facultatifs)

Élément de langage	Mot-clé	Détails
déclaration de bloc fonction	VAR	contient les variables locales
fonction d'appartenance	variable d'entrée:TERM	maximum de quatre points constants ou variables (coordonnée de degré d'appartenance = 0 ou 1)
	variable de sortie:TERM	maximum de quatre points constants ou variables (coordonnée de degré d'appartenance = 0 ou 1)
agrégation conditionnelle	opérateur: AND	algorithme: PROD , BDIF
	opérateur: OR	algorithme: ASUM , BSUM
	opérateur: NOT	1 – {argument}
	parenthèses	()
activation	opérateur: ACT	algorithme: MIN, PROD
accumulation	opérateur: ACCU	algorithme: BSUM , NSUM
intervalle de fuzzification	opérateur: RANGE	RANGE (valeur minimum, valeur maximum), limite l'intervalle des fonctions d'appartenance pour la variable de sortie.
méthode de défuzzification	opérateur: METHOD	algorithme: CoG , CoA , LM , RM
bloc de règles	opérateur: RULEBLOCK	n blocs de règles
condition	IF	n sous-conditions, n variables d'entrée
conclusion	THEN	n sous-conclusions, n variables de sortie
facteur de pondération	WITH	valeur constante et valeur assignée à une variable dans la section de déclaration VAR_INPUT.....END_VAR

Le tableau 10 contient un exemple de liste d'éléments de langage de niveau ouvert. Cette liste doit accompagner la documentation du système.

Tableau 10 – Exemple de liste d'éléments de langage Niveau Ouvert

fonctions d'appartenance à entrée/sortie libre (ex. Gaussien, exponentiel)
plus de quatre points de fonction d'appartenance
valeurs des coordonnées de degré d'appartenance entre 0 et 1
valeurs d'appartenance variable

6.2 Liste de contrôle de données

Cette liste de contrôle de données (voir tableau 11) doit être fournie avec la documentation technique. Dans cette liste, tout fabricant d'automates programmables et d'outils de programmation et logiciels d'application de contrôle flou doit décrire les caractéristiques de performance spécifiques de son système de contrôle flou. Pour faciliter le transfert des applications de contrôle flou entre systèmes produits par différents fabricants, la liste de contrôle de données suivante constitue un moyen de vérifier le transfert possible d'un programme. Cette liste non exhaustive est fournie à titre d'exemple seulement et peut être complétée par les fabricants.

Table 9 – FCL Extension Level language elements (optional)

Language element	Keyword	Details
function block declaration	VAR	contains local variables
membership function	input variable:TERM	maximum of four constant or variable points (degree of membership co-ordinate = 0 or 1)
	output variable:TERM	maximum of four constant or variable points (degree of membership co-ordinate = 0 or 1)
conditional aggregation	operator: AND	algorithm: PROD , BDIF
	operator: OR	algorithm: ASUM , BSUM
	operator: NOT	1 – {argument}
	parentheses	()
activation	operator: ACT	algorithm: MIN, PROD
accumulation	operator: ACCU	algorithm: BSUM , NSUM
range for fuzzification	operator: RANGE	RANGE (minimum value..maximum value), limits the range of the membership functions for the output variable
defuzzification method	operator: METHOD	algorithm: CoG , CoA , LM , RM
ruleblock	operator: RULEBLOCK	n rule blocks
condition	IF	n subconditions, n input variables
conclusion	THEN	n subconclusions, n output variables
weighting factor	WITH	constant value and value assigned to variable in the declaration part VAR_INPUT.....END_VAR

The table 10 shows an example of a list of language elements in the Open Level. This list shall be a part of the system documentation.

Table 10 – Examples of a list with Open Level language elements

free input/output membership functions (e.g. Gaussian, exponential)
more than four membership function points
degree of membership co-ordinate values from 0 to 1
variable membership values

6.2 Data check list

This data check list (see table 11) shall be delivered within the technical documentation. In this list, a manufacturer of programmable controllers, fuzzy control programming tools and application software shall describe specific performance features of the fuzzy control system. In order to facilitate the transfer of fuzzy control applications among different manufacturers' systems, the following Data check list is the means of verifying a possible program transfer. This list is not intended to be comprehensive but serves as an example only and could be enhanced by manufacturers.

Tableau 11 – Liste de contrôle de données

Données techniques	Déclaration du fabricant (exemples)
types de données des entrées et sorties de bloc fonction	REEL, ENT
commentaires de lignes de code dans le programme FCL	OUI, NON
temps d'exécution (ms)	20, 30
besoins en mémoire (kb)	3, 4
transposition des valeurs variables de facteurs de pondération et degrés d'appartenance (qui varient entre 0,0 et 1,0) sur une échelle de nombres entiers	0-200, 0-400
longueur des identifiants (ex. noms attribués aux variables, blocs de règles, termes)	6, 8
nombre maximal de variables d'entrée pour la fuzzification	6, 8
nombre maximal de termes de fonction d'appartenance par variable d'entrée	5, 7
nombre total maximal de termes de fonction d'appartenance pour toutes les variables d'entrée	30, 56
nombre total maximal de points pour la fonction d'appartenance associée à chaque terme de variable d'entrée	3, 4, 10
nombre total maximal de points pour les fonctions d'appartenance associées à tous les termes de variable d'entrée	90, 224
nombre maximal de variables de sortie pour la défuzzification	6, 8
nombre maximal de termes de fonction d'appartenance par variable de sortie	5, 7
nombre total maximal de termes de fonction d'appartenance pour toutes les variables de sortie	30, 56
nombre maximal de points pour la fonction d'appartenance associée à chaque terme de variable de sortie	1, 4, 10
nombre total maximal de points pour les fonctions d'appartenance associées à tous les termes de variable de sortie	90, 224
nombre maximal de blocs de règles	1, 10
nombre maximal de règles par bloc	10
nombre maximal de sous-conditions par règle	4, 10
nombre total maximal de règles	15
nombre maximal de sous-conclusions par règle	4
profondeur de récursion (r)	1, 3

Table 11 – Data check list

Technical data	Manufacturer statement (examples)
data types of function block inputs and outputs	<i>REAL, INT</i>
line comments in the FCL program	<i>YES, NO</i>
execution time (ms)	20, 30
memory requirements (kb)	3, 4
mapping of the variable values of weighting factors and membership degrees from 0,0 to 1,0 onto the range of integer values	0-200, 0-400
length of identifiers (e.g. name of variables, rule blocks, terms)	6, 8
max. number of input variables for fuzzification	6, 8
max. number of membership function terms per input variable	5, 7
max. total number of membership function terms for all input variables	30, 56
max. number of points for the membership function associated with each input variable term	3, 4, 10
max. total number of points for membership functions associated with all input variable terms	90, 224
max. number of output variables for defuzzification	6, 8
max. number of membership function terms per output variable	5, 7
max. total number of membership function terms for all output variables	30, 56
max. number of points for the membership function associated with each output variable term	1, 4, 10
max. total number of points for membership functions associated with the all output variable terms	90, 224
max. number of rule blocks	1, 10
max. number of rules per block	10
max. number of subconditions per rule	4, 10
max. number of all rules	15
max. number of subconclusions per rule	4
nesting depth of ()	1, 3

Annexe A (informative)

Théorie

Cette annexe est une explication des définitions de l'article 3.

A.1 Logique floue

En *logique floue*, les variables physiques sont décrites par des valeurs et des expressions linguistiques, qui remplacent les noms et les nombres (généralement réels) utilisés dans les systèmes de contrôle à boucle ouverte ou fermée conventionnels. Les termes «basse» ou «grande ouverte» sont désignés comme des *termes linguistiques*, respectivement associés aux valeurs physiques de «température» et de «position de vanne de chauffage». Une variable d'entrée décrite par des *termes linguistiques* prend le nom de *valeur linguistique*.

Chaque *terme linguistique* est décrit par un *ensemble flou* M , et est donc mathématiquement défini sans équivoque par deux propositions, celle de l'ensemble de base G et celle de la *fonction d'appartenance* μ . La *fonction d'appartenance* précise le degré d'appartenance de chaque élément de l'univers de discours G (par exemple par une valeur numérique sur une échelle de temps [années d'âge]) à l'ensemble M (par exemple «jeune»), sous la forme d'une valeur numérique comprise entre zéro et un. Si la *fonction d'appartenance* pour une valeur spécifique est égale à un, alors l'énoncé linguistique correspondant au *terme linguistique* s'applique totalement («jeune» pour un âge de 20 ans, par exemple). Par contre, si la fonction d'appartenance est égale à zéro, aucun accord n'existe entre les deux («très vieux» pour un âge de 20 ans, par exemple).

La description des *ensembles flous* fait appel aux notations suivantes:

pour ensembles finis: ensembles appariés, non ordonnés sous forme incrémentielle:

$$M = \{(x_1, \mu_M(x_1)), (x_2, \mu_M(x_2)), \dots, (x_n, \mu_M(x_n))\}, x_i \in G, i = 1, 2, \dots, n \quad (A.1)$$

les $\mu_M(x_i)$ s'affichent en valeurs numériques.

pour ensembles infinis:

$$M = \{x, \mu_M(x)\}, x \in G \quad (A.2)$$

Pour mieux illustrer les différences entre termes *réels* et termes *flous*, les *termes linguistiques* de la *variable linguistique* «Age» sont représentés à la figure A.1. Alors que «l'âge adulte légal» est stipulé sans équivoque par les lois, et donc présente une transition discrète par rapport à la *fonction d'appartenance*, il est impossible de définir une limite d'âge réelle pour le terme «adulte».

Annex A (informative)

Theory

This annex is an explanation of the definitions given in clause 3.

A.1 Fuzzy Logic

In *fuzzy logic*, linguistic values and expressions are used to describe physical variables, instead of the names, numbers (usually real numbers) used in conventional open and closed-loop control systems. The terms "low" or "wide-open" are designated as *linguistic terms* of the physical values "temperature" or "heating valve opening". If an input variable is described by *linguistic terms*, it is referred to as a *linguistic value*.

Each *linguistic term* is described by a *fuzzy Set* M. It is thus unequivocally defined mathematically by the two statements basic set G and *membership function* μ . The *membership function* states the membership of every element of the universe of discourse G (e.g. numerical values of a time scale [age in years]) in the set M (e.g. "young") in the form of a numerical value between zero and one. If the *membership function* for a specific value is one, then the linguistic statement corresponding to the *linguistic term* applies in all respects (e.g. "young" for an age of 20 years). If, in contrast, it is zero, then there is absolutely no agreement (e.g. "very old" for an age of 20 years).

The following notations are used to describe *fuzzy sets*:

for finite sets: as unordered, paired sets in incremental form:

$$M = \{(x_1, \mu_M(x_1)), (x_2, \mu_M(x_2)), \dots, (x_n, \mu_M(x_n))\}, \quad x_i \in G, i=1,2,\dots,n \quad (A.1)$$

the $\mu_M(x_i)$ are listed as numerical values.

for infinite sets:

$$M = \{x, \mu_M(x)\}, \quad x \in G \quad (A.2)$$

In order to more clearly illustrate the differences between *crisp* and *fuzzy* terms, the *linguistic terms* of the *linguistic variable* "Age" are represented in figure A.1. While "full legal age" is unequivocally stipulated by law, and thus displays a discrete transition in relation to the *membership function*, a crisp age limit may not be given for "adult".

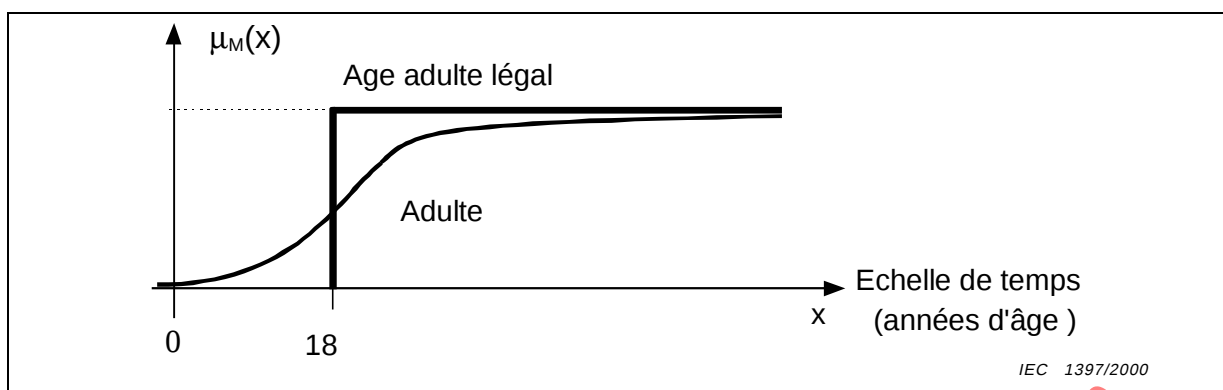


Figure A.1 – Fonctions d'appartenance pour les termes «âge adulte légal» et «adulte»

L'exemple illustré à la figure A.2 présente une description de la *variable linguistique* «Age» par des *termes linguistiques* et leur hiérarchie sur l'échelle de temps «années d'âge» au moyen de *valeurs d'appartenance*.

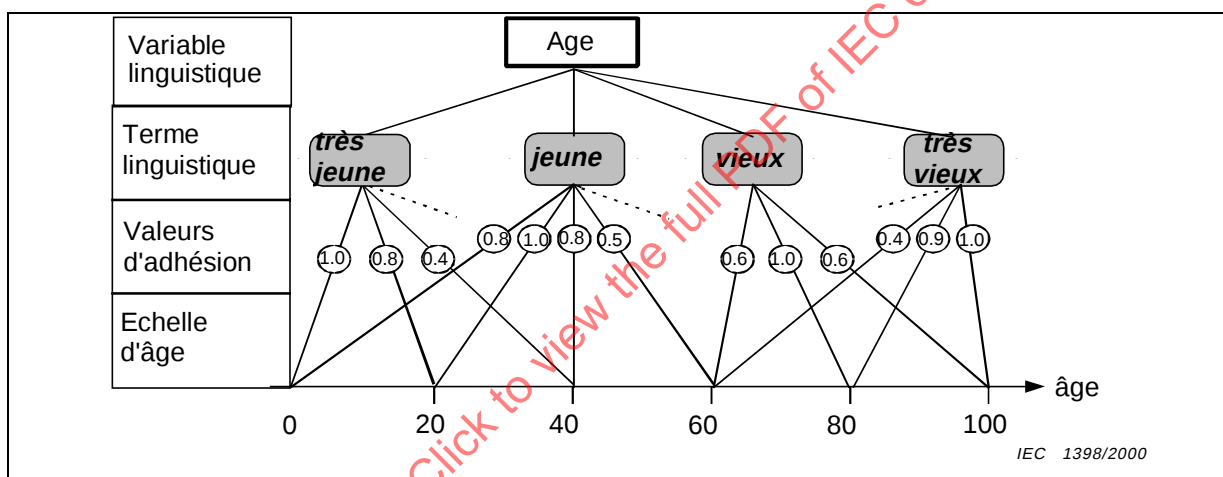


Figure A.2 – Description de la variable linguistique «Age» par des termes linguistiques et par leur hiérarchie sur l'échelle de temps (années d'âge)

Des profils typiques de *fonctions d'appartenance* sont représentés à la figure A.3. Les profils suivants sont des profils spéciaux:

- la définition par un rectangle (ex. intervalle) pour décrire des *valeurs*, ainsi que
- le «*singleton*», pour la représentation alternative des variables de sortie.



Figure A.1 – Membership functions of the terms "full legal age" and "adult"

As an example, figure A.2 shows the description of the *linguistic variable* "Age" by *linguistic terms* and their hierarchy on the time scale "age in years" by means of *membership values*.

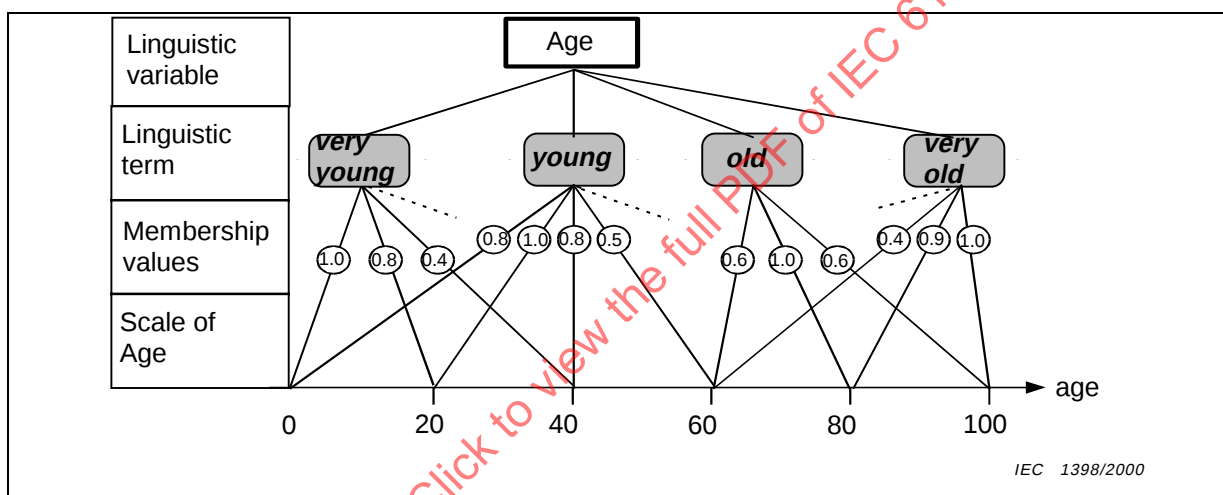


Figure A.2 – Description of the linguistic variable "Age" by linguistic terms and their hierarchy on the time scale (age in years)

Typical forms of the *membership functions* are represented in figure A.3. The following count as special forms:

- the definition via the rectangle (e.g. interval) in order to describe *values*, as well as
- the "*singleton*", for the alternative representation of output variables.

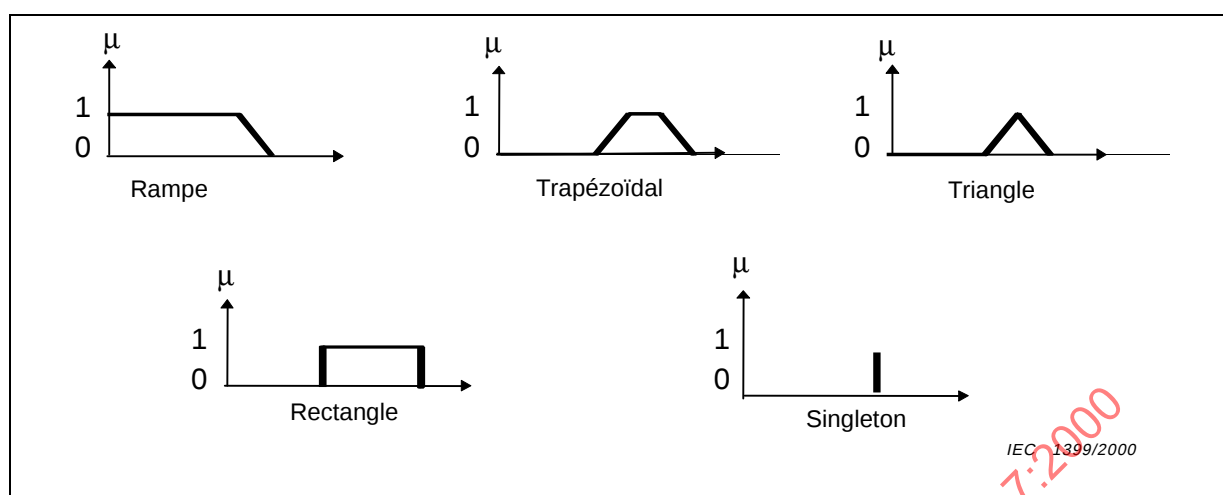


Figure A.3 – Profils de fonctions d'appartenance fréquemment utilisés

Une expression dans laquelle des *variables linguistiques* sont associées à des *termes linguistiques* représente un *énoncé linguistique en logique floue*. Les expressions telles que «La température est élevée» ou «La température est basse», dont la structure de base est

<i>Variable linguistique – Symbole de comparaison – Terme linguistique</i> (La température est basse)	(A.3)
--	-------

sont des énoncés linguistiques.

Contrairement à la logique classique, selon laquelle une affirmation est soit «vraie» soit «fausse» (un des deux états Booléens), un énoncé linguistique en logique floue possède un degré d'appartenance.

La connaissance empirique peut être décrite par des règles. La règle R_k prend la forme suivante:

R_k : IF <i>condition</i> P_k THEN <i>conclusion</i> C_k	(A.4)
--	-------

Dans ce contexte, la *condition* de chaque *règle* comporte un énoncé linguistique ou une combinaison d'énoncés définis par les variables d'entrée, alors que la *conclusion* fournit une variable de sortie sous la forme d'une instruction d'action spécifique.

$P_k = A \text{ AND } B \text{ OR } (\text{NOT } C)$	(A.5)
--	-------

Si la *condition* et/ou la *conclusion* sont déterminées par des *énoncés linguistiques*, la *règle* est alors également appelée *règle linguistique*. Par ailleurs, une *base de règles* est un regroupement de plusieurs *règles*. En général, plusieurs *règles* s'appliquent (sont «activées») en même temps, contrairement aux systèmes régis par des règles classiques. Par conséquent, il faut que les résultats des règles soient combinés entre eux par les *opérateurs* mathématiques correspondants.

Les relations entre les *ensembles flous* et les *opérations sur ensembles flous* sont définies par les *fonctions d'appartenance*:

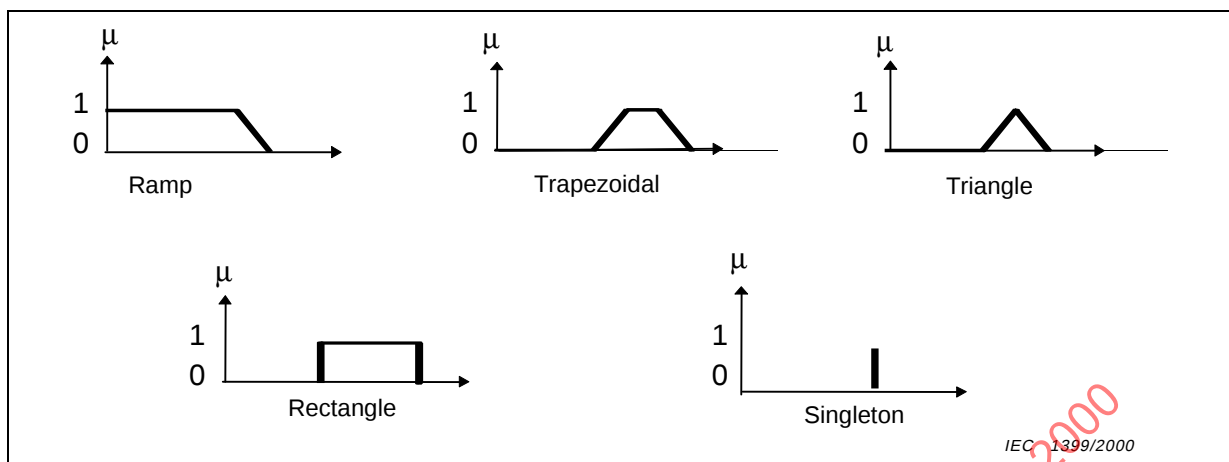


Figure A.3 – Commonly used shapes of membership functions

An expression in which *linguistic variables* are related to *linguistic terms* represents a *linguistic statement* in *fuzzy logic*. Expressions such as "Temperature is high" or "Temperature is low", with the simple basic structure of

<i>Linguistic variable</i> – <i>Symbol of comparison</i> – <i>Linguistic term</i> (Temperature is low)	(A.3)
--	-------

are referred to here as linguistic statement.

In contrast to classical logic, in which statements only assume one of the Boolean states "true" or "false", linguistic statements in fuzzy logic possess a degree of membership.

Empirical knowledge may be defined in rules. Rule R_k has the following form:

R_k : IF <i>condition</i> P_k THEN <i>conclusion</i> C_k	(A.4)
--	-------

In this context, the *condition* of each *rule* comprises a linguistic statement or a combination of statements via the input variables, while the *conclusion* determines the output variable in the sense of an instruction to act.

$P_k = A \text{ AND } B \text{ OR } (\text{NOT } C)$	(A.5)
--	-------

If *condition* and/or *conclusion* are determined by *linguistic statements*, the *rule* is then also referred to as a *linguistic rule*. A *rule base*, in turn, consists of several *rules* together. In general, several *rules* apply ("fire") at the same time in contrast to classical rule-based systems. Therefore, the results of the rules must be combined with one another via corresponding mathematical *operators*.

Relationships between *fuzzy sets* and *operations* with *fuzzy sets* are defined by the *membership functions*:

Les relations suivantes s'appliquent entre deux *ensembles flous*, A et B, dont les éléments x proviennent d'un ensemble de base G:

égalité $A = B$,
est vraie si $\mu_A(x) = \mu_B(x)$, quel que soit $x \in G$ (A.6)

inclusion complète $A \subseteq B$,
est vraie si $\mu_A(x) \leq \mu_B(x)$, quel que soit $x \in G$ (A.7)

inclusion partielle $A \subset B$,
est vraie si $\mu_A(x) \leq \mu_B(x)$, quel que soit $x \in G$
et $\mu_A(x) < \mu_B(x)$ pour au moins un $x \in G$. (A.8)

Les *opérations* suivantes peuvent être stipulées entre deux *ensembles flous*, A et B, dont les éléments x proviennent d'un ensemble de base G:

l'intersection
 $A \cap B$ est définie par $\mu_{A \cap B}(x) = I(\mu_A(x), \mu_B(x))$, (A.9)

où I représente l'opérateur d'intersection

l'union
 $A \cup B$ est définie par $\mu_{A \cup B}(x) = U(\mu_A(x), \mu_B(x))$, (A.10)

où U représente l'opérateur d'union

le *complément* est défini par $\mu_A^-(x) = 1 - \mu_A(x)$ (A.11)

Comme pour les ensembles «nets» (classiques), les interprétations suivantes s'appliquent aux ensembles flous:

l'intersection est associée à l'opérateur flou ET,

l'union est associée à l'opérateur flou OU,

le *complément* est associé à l'opérateur flou PAS.

Les algorithmes élémentaires pour l'implémentation mathématique d'intersection, union et complément sont les suivants (voir aussi figure A.4):

pour *l'intersection*, le minimum
 $\mu_{A \cap B}(x) = \text{Min} \{ \mu_A(x), \mu_B(x) \}, x \in G$ (A.12)

pour *l'union*, le maximum
 $\mu_{A \cup B}(x) = \text{Max} \{ \mu_A(x), \mu_B(x) \}, x \in G$ (A.13)

pour le *complément*, soustraction de un
 $\mu_A^-(x) = 1 - \mu_A(x) \quad x \in G$ (A.14)

De nombreuses possibilités d'algorithmes existent pour les opérateurs flous ET et OU, mais il convient de mentionner que les opérateurs ET et OU ne peuvent pas être choisis de manière arbitraire.

The following relationships apply between two *fuzzy sets*, A and B, whose elements x originate from a basic set G:

equality $A = B$,

$$\text{is true if } \mu_A(x) = \mu_B(x), \text{ for all } x \in G \quad (\text{A.6})$$

complete inclusion $A \subseteq B$,

$$\text{is true if } \mu_A(x) \leq \mu_B(x), \text{ for all } x \in G \quad (\text{A.7})$$

partial inclusion $A \subset B$,

$$\begin{aligned} &\text{is true if } \mu_A(x) \leq \mu_B(x), \text{ for all } x \in G \\ &\text{and } \mu_A(x) < \mu_B(x) \quad \text{for at least one } x \in G. \end{aligned} \quad (\text{A.8})$$

The following *operations* may be stipulated between two *fuzzy sets*, A and B, whose elements x originate from a basic set G:

the intersection

$$A \cap B \text{ is defined by } \mu_{A \cap B}(x) = I(\mu_A(x), \mu_B(x)), \quad (\text{A.9})$$

where I is called the intersection operator

the union

$$A \cup B \text{ is defined by } \mu_{A \cup B}(x) = U(\mu_A(x), \mu_B(x)), \quad (\text{A.10})$$

where U is called the union operator

the complement

$$\text{is defined by } \mu_A^-(x) = 1 - \mu_A(x) \quad (\text{A.11})$$

As with classical (*crisp*) sets, the following interpretations apply to fuzzy sets:

the *intersection* is related to the fuzzy operator AND,
the *union* is related to the fuzzy operator OR,
the *complement* is related to the fuzzy operator NOT.

Elementary algorithms for mathematical implementation of intersection, union and complement are the following (see also figure A.4):

for the *intersection*, the minimum

$$\mu_{A \cap B}(x) = \text{Min } \{\mu_A(x), \mu_B(x)\}, \quad x \in G \quad (\text{A.12})$$

for the *union*, the maximum

$$\mu_{A \cup B}(x) = \text{Max } \{\mu_A(x), \mu_B(x)\}, \quad x \in G \quad (\text{A.13})$$

for the *complement*, subtraction from one

$$\mu_A^-(x) = 1 - \mu_A(x) \quad x \in G \quad (\text{A.14})$$

There are a lot of possible algorithms for AND and OR fuzzy operators. It is worth noting here, that AND and OR operators cannot be chosen in an arbitrary way.

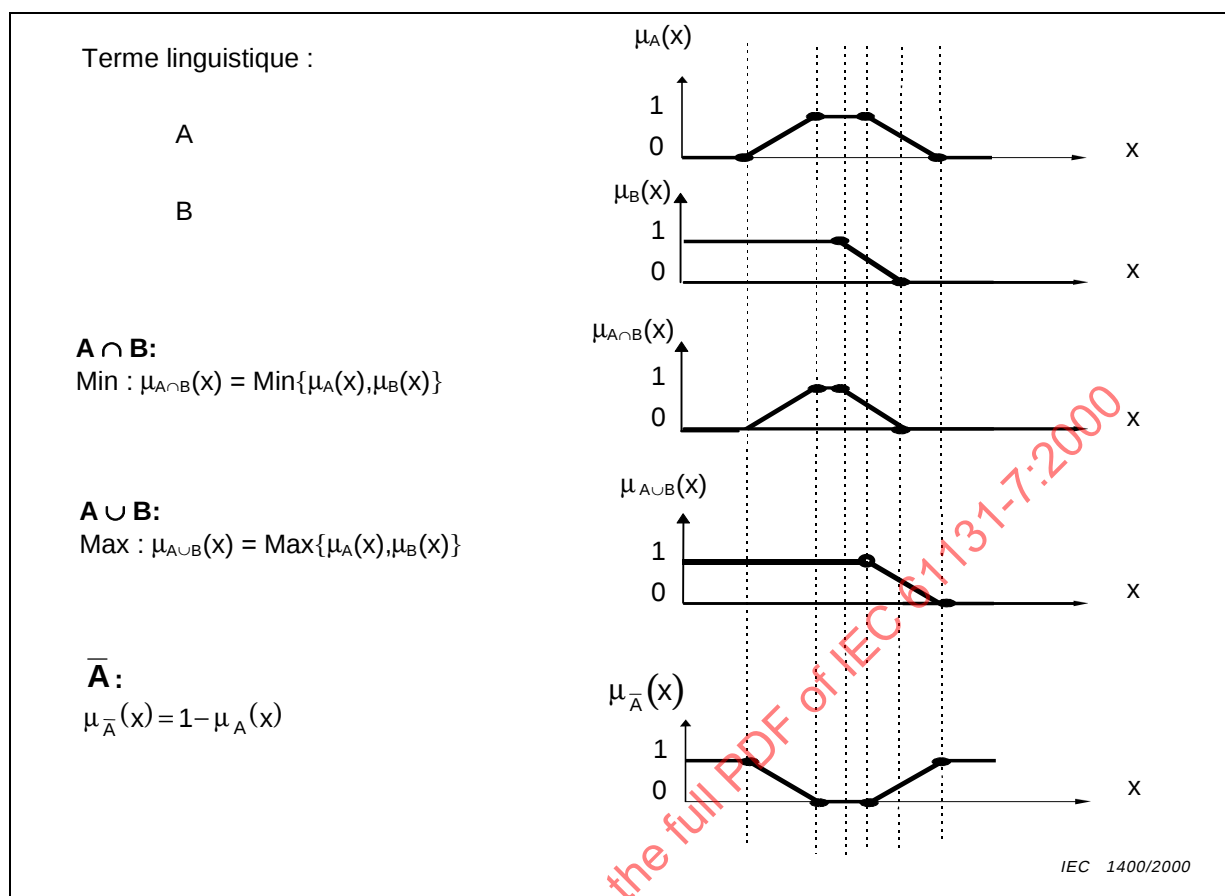


Figure A.4 – Algorithmes pour l'implémentation d'opérations entre deux fonctions d'appartenance

A.2 Contrôle flou

Le contrôle flou se traduit par la possibilité de contrôler des processus techniques en boucle ouverte ou fermée, y compris le traitement de valeurs mesurées, au moyen de règles floues et de leur traitement par logique floue.

Les informations d'entrée comportent des variables réelles sous forme de variables de processus mesurables, variables dérivées et points de consigne. Les variables de sortie sont des variables réelles sous forme de variables de correction. Il faut que des transformations se produisent entre les variables d'entrée et de sortie du processus et le monde flou (fuzzification, défuzzification). Les règles linguistiques de la base de règles et l'inférence constituent l'élément de base du contrôle flou.

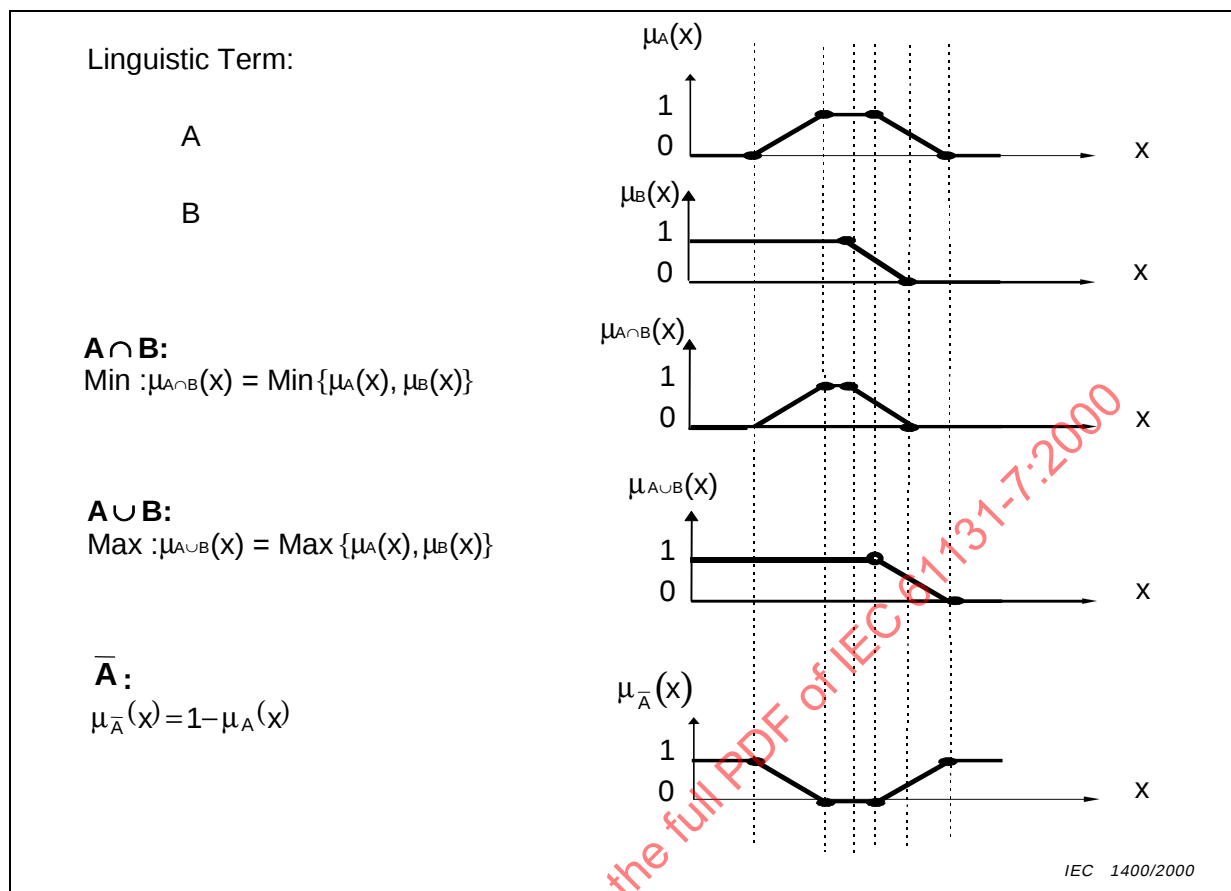


Figure A.4 – Algorithms for implementing operations between two membership functions

A.2 Fuzzy Control

Fuzzy control means the open and closed-loop control of technical processes, including the processing of measured values, which is based on the use of fuzzy rules and their processing with the help of fuzzy logic.

The input information comprises real variables in the form of measurable process variables, derived variables, as well as set points. The output variables are real variables in the form of correcting variables. Transformations must be performed between the input and output variables of the process and the fuzzy world (fuzzification, defuzzification). The core component of fuzzy control consists of the linguistic rules of the rule base and the inference.

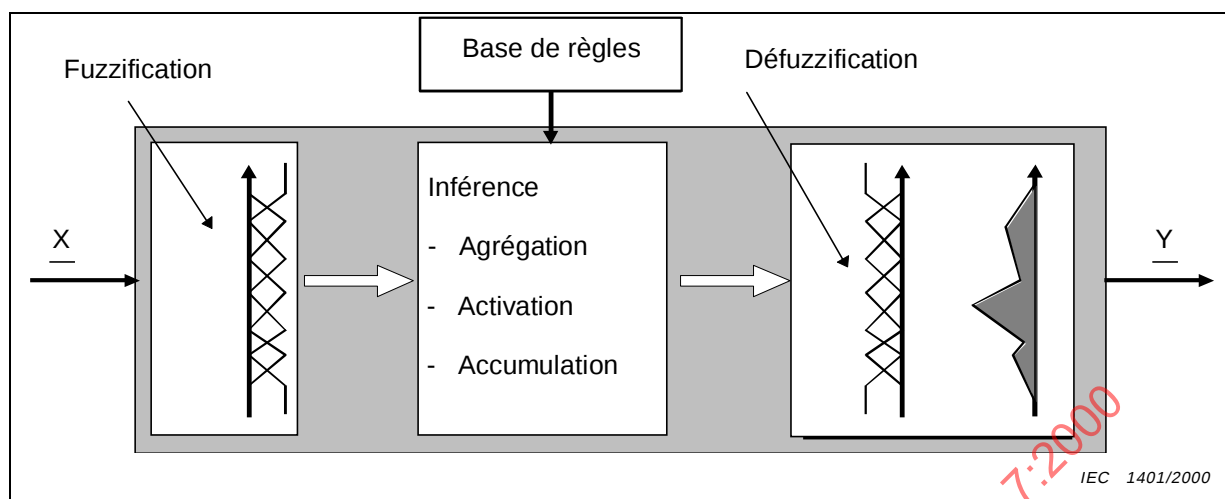


Figure A.5 – Structure et éléments fonctionnels du contrôle flou

Les éléments fonctionnels de *Contrôle flou*, évoqués plus haut et représentés à la figure A.5, sont décrits ci-dessous.

A.2.1 Fuzzification

Le calcul de la correspondance entre les variables d'entrée et les *termes linguistiques* est appelé *fuzzification*. Pour ce calcul, le degré d'appartenance réel des variables d'entrée est déterminé pour chaque *terme linguistique* de la *variable linguistique* correspondante.

Un exemple de *fuzzification* est illustré à la figure A.6.

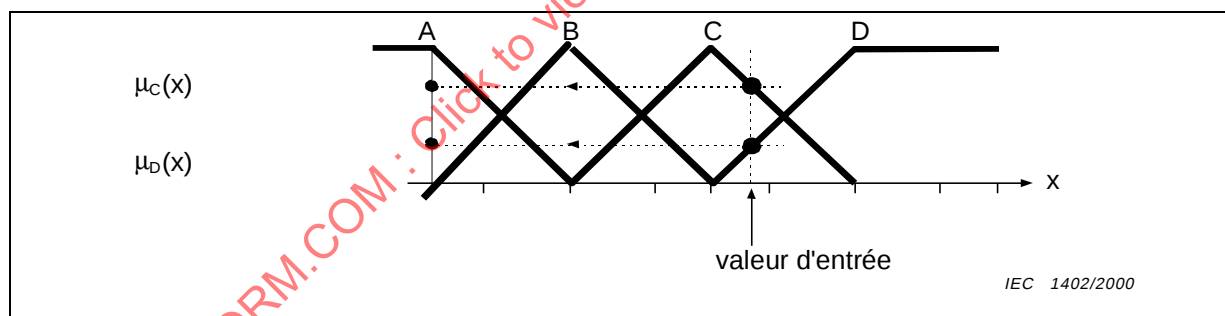


Figure A.6 – Principe de fuzzification (exemple)

A.2.2 Base de règles

La *base de règles* contient la connaissance empirique associée à l'exécution d'un processus particulier. Des *règles linguistiques* sont utilisées pour représenter la connaissance. Si l'opérateur flou ET correspond à MIN et l'opérateur flou OU à MAX, il est facilement démontré qu'une règle floue R_j basée sur une combinaison OU de m énoncés peut être représentée par m règles, dont les énoncés ne sont combinés que par des ET. Des exemples de différentes représentations de règles sont présentés à la figure A.7 et à la figure A.8.

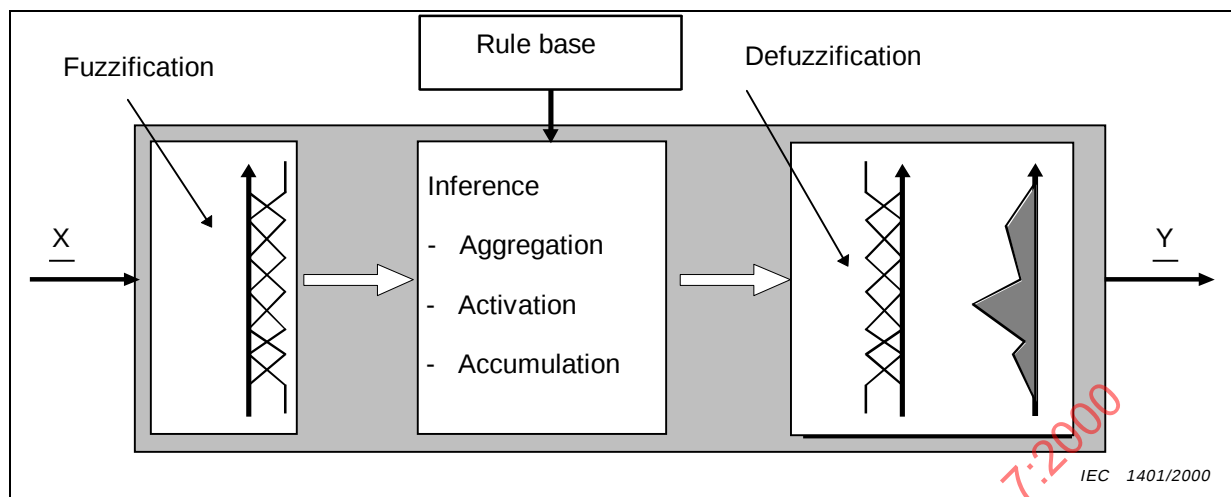


Figure A.5 – Structure and functional elements of fuzzy control

The functional elements of *fuzzy control*, mentioned above and represented in figure A.5, are explained below.

A.2.1 Fuzzification

The determination of the matching of input variables with the *linguistic terms* is referred to as *fuzzification*. To this end, the actual degree of membership for input variables is determined for each *linguistic term* of the corresponding *linguistic variable*.

Figure A.6 shows an example of *fuzzification*.

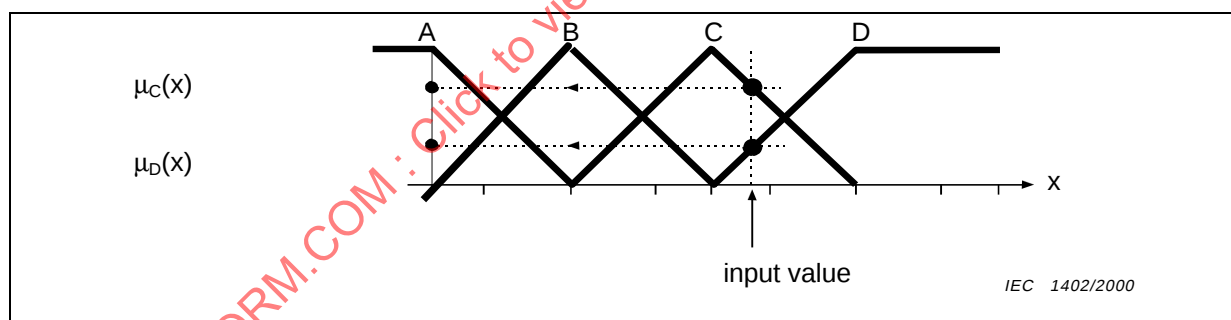


Figure A.6 – The principle of fuzzification (as an example)

A.2.2 Rule base

The *rule base* contains empirical knowledge concerning the operation of a particular process under consideration. *Linguistic rules* are used to represent the knowledge. Provided that the AND fuzzy operator is MIN and the OR fuzzy operator is MAX, it can easily be shown that a fuzzy rule R_j based on an OR combination of m statements may be represented by m rules, whose statements are only combined by AND. Figure A.7 and figure A.8 are examples of different rule representations.

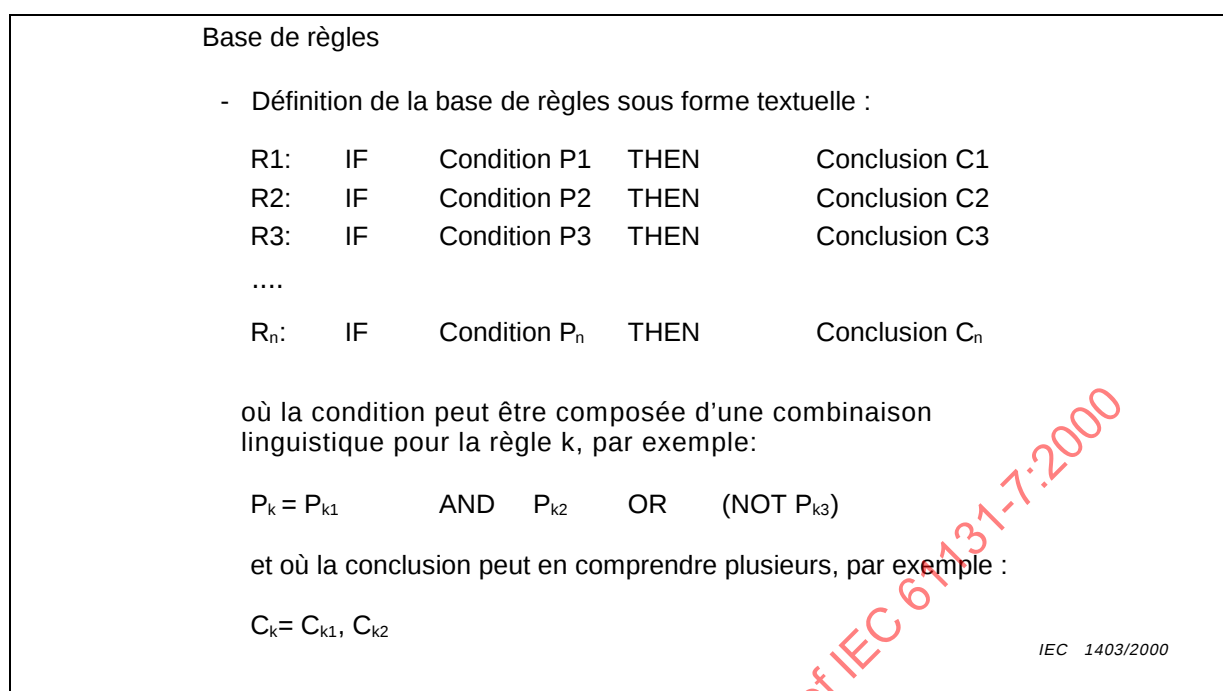


Figure A.7 – Représentation de la base de connaissance sous forme linguistique

Lorsque deux variables d'entrée et une variable de sortie sont disponibles, et si ces deux variables d'entrée ne sont combinées que par ET, la *base de règles* peut être donnée sous forme de matrice; les valeurs de variable d'entrée sont assignées aux colonnes et/ou lignes, et les champs de la matrice contiennent les valeurs de la variable de sortie.

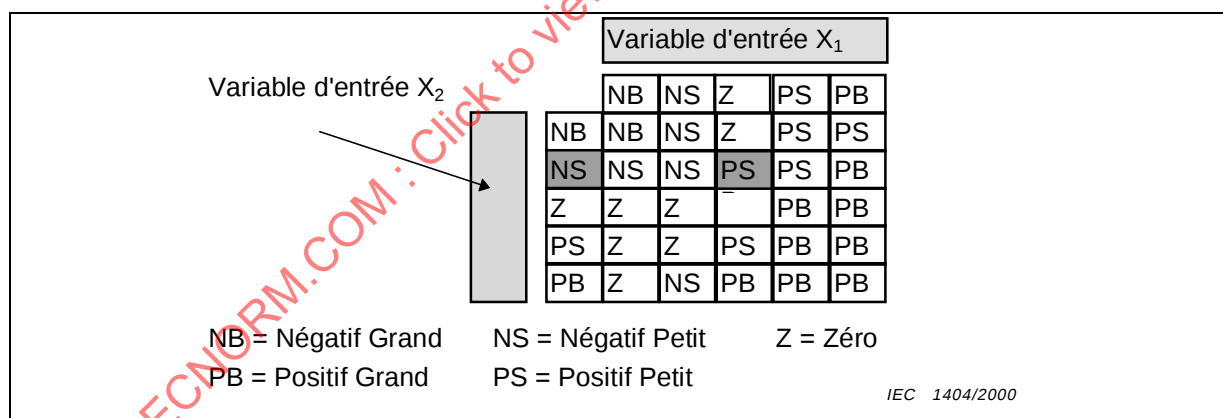


Figure A.8 – Représentation de deux variables sous forme de matrice

A.2.3 Inférence

Les principes et les définitions utilisés ici sont valides pour un cas simplifié de base de règles floues, qui s'inspire essentiellement du mécanisme d'inférence couramment utilisé de Mamdani. Les mécanismes d'inférence plus complexes ne sont pas examinés ici.

L'*Inférence* est composée des trois sous-fonctions *agrégation*, *activation* et *accumulation* représentées à la figure A.9. Voir aussi la figure A.10.

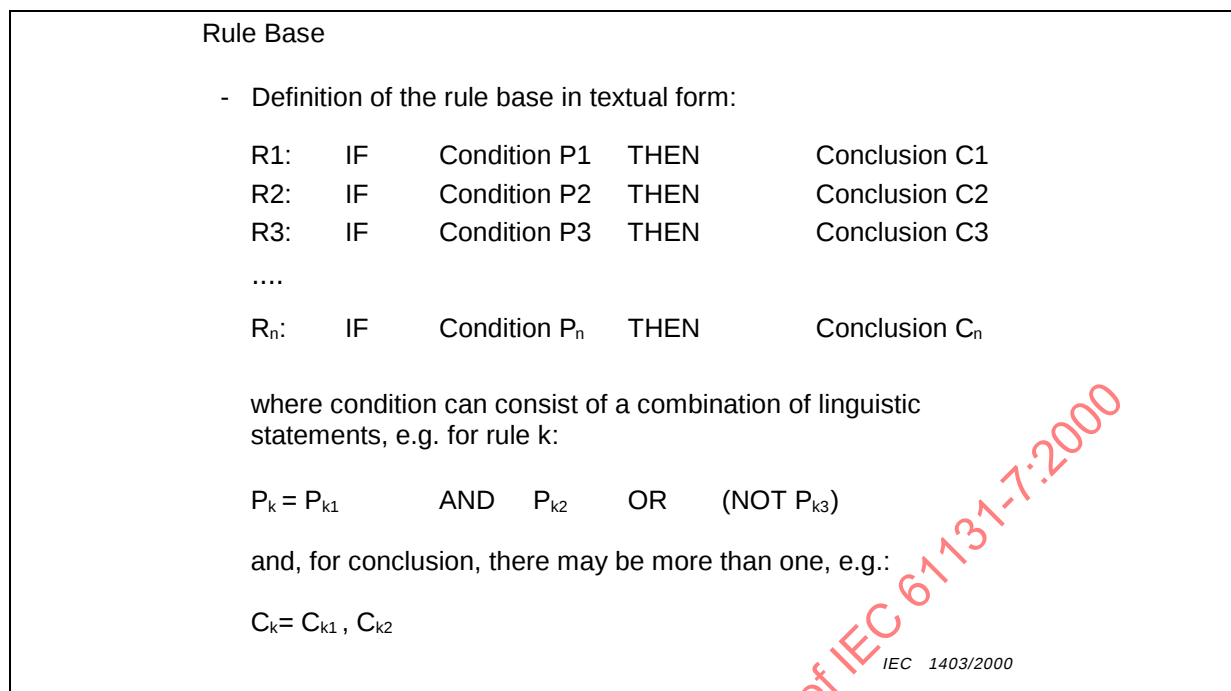


Figure A.7 – Representation of the knowledge base in linguistic form

If two input variables and one output variable are available, and if these two input variables are combined only by AND, the *rule base* may be given in the form of a matrix, where the values of the input variables are assigned to the columns and/or lines, and the fields of the matrix contain the values of the output variable.

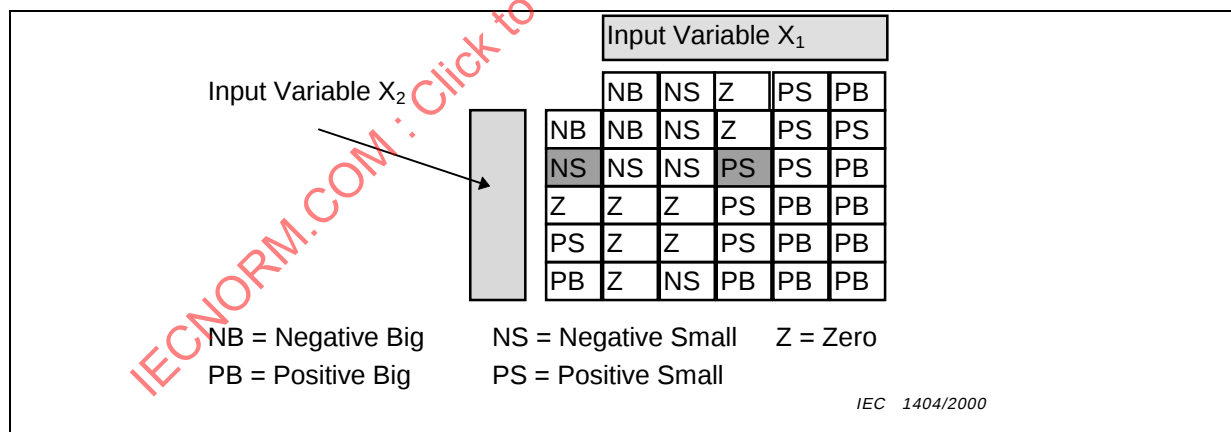


Figure A.8 – Matrix representation of two variables

A.2.3 Inference

The principles and definitions used here are valid for a simplified case of fuzzy rule base, basically inspired from the widely used Mamdani inferencing scheme. More complicated inferencing schemes are not discussed here.

The *Inference* consists of the three subfunctions *aggregation*, *activation* and *accumulation* shown in figure A.9. See also figure A.10.

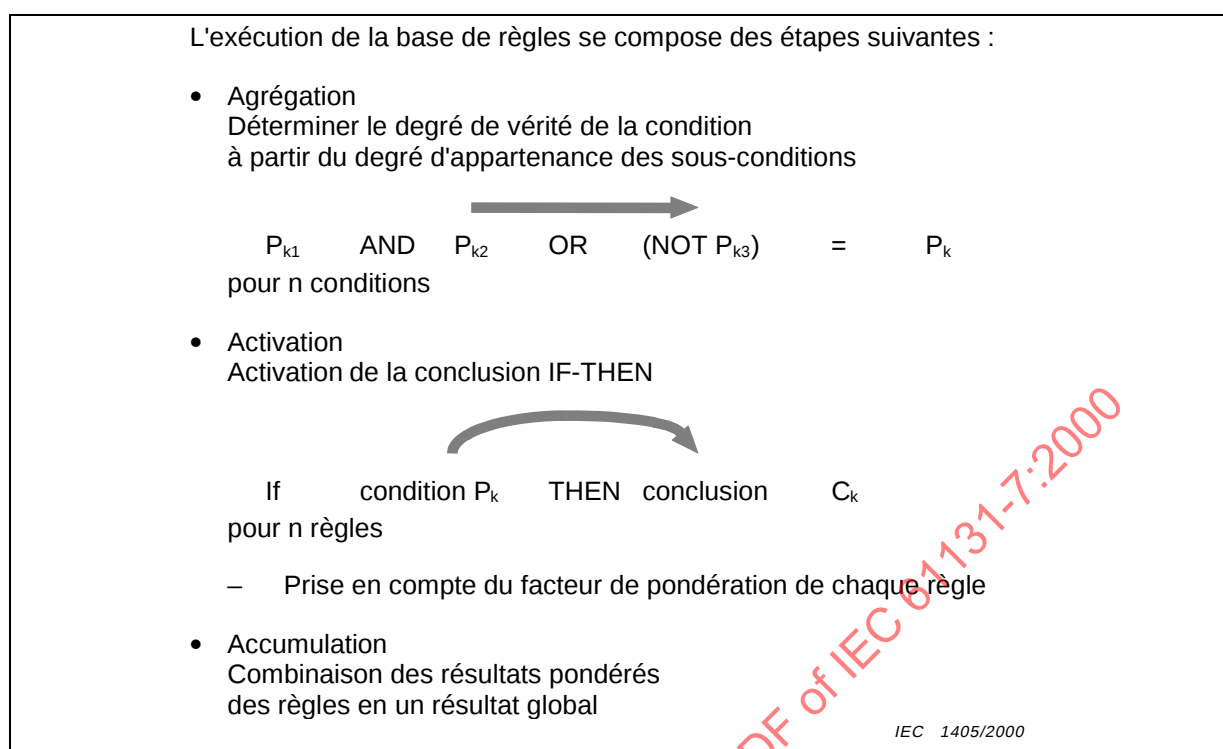


Figure A.9 – Eléments d'une inférence

– **Agrégation:**

Si la condition se compose d'une seule sous-condition, alors la validité de la *condition* est identique à celle de la sous-condition, c'est-à-dire que le degré de vérité de la *condition* correspond à la sous-condition. Cependant, si la *condition* se compose d'une combinaison de plusieurs sous-conditions, il faut que le degré de vérité soit déterminé par *agrégation* des différentes valeurs. Dans le cas de sous-conditions combinées par des ET, le degré de vérité est calculé au moyen de l'opérateur flou ET.

– **Activation:**

Dans la *conclusion*, les *sous-conclusions* sont associées aux variables de sortie. Le *degré d'appartenance* de la *conclusion* est ensuite déterminé à partir du *degré de vérité* de la *condition* déterminée par agrégation (*Conclusion* SI A ALORS B). Les opérateurs MIN ou PROD sont généralement utilisés pour l'activation.

Si la base de règles contient des règles à facteurs de pondération w_k , où $w_k \in [0, 1]$, ceux-ci peuvent être implémentés par multiplication.

$$c_k^* = w_k \times c_k \quad (\text{A.15})$$

– **Accumulation:**

Les résultats des *règles* sont combinés en un résultat global. L'accumulation fait généralement appel à l'*algorithme maximum*. Les *opérateurs* généralement utilisés aux différentes étapes d'*inférence* sont présentés au tableau A.1.

Suivant la combinaison d'*opérateurs* des différentes étapes, différentes stratégies d'*inférence* sont possibles. Les plus connues sont l'*inférence MaxMin* et l'*inférence MaxProd*, qui utilisent le *maximum* pour l'*accumulation* et le *minimum* ou le *produit algébrique* pour l'*activation*. Dans le cas de l'*inférence MaxMin*, les fonctions d'appartenance des *ensembles flous* des *conclusions* sont limitées au *degré de vérité* de la condition, puis combinées pour créer un *ensemble flou* en formant un maximum. Dans l'*inférence MaxProd*, en revanche, les *fonctions d'appartenance* des *ensembles flous* des *conclusions* sont pondérées, c'est-à-dire multipliées par le *degré de vérité* de la *condition* puis combinées.

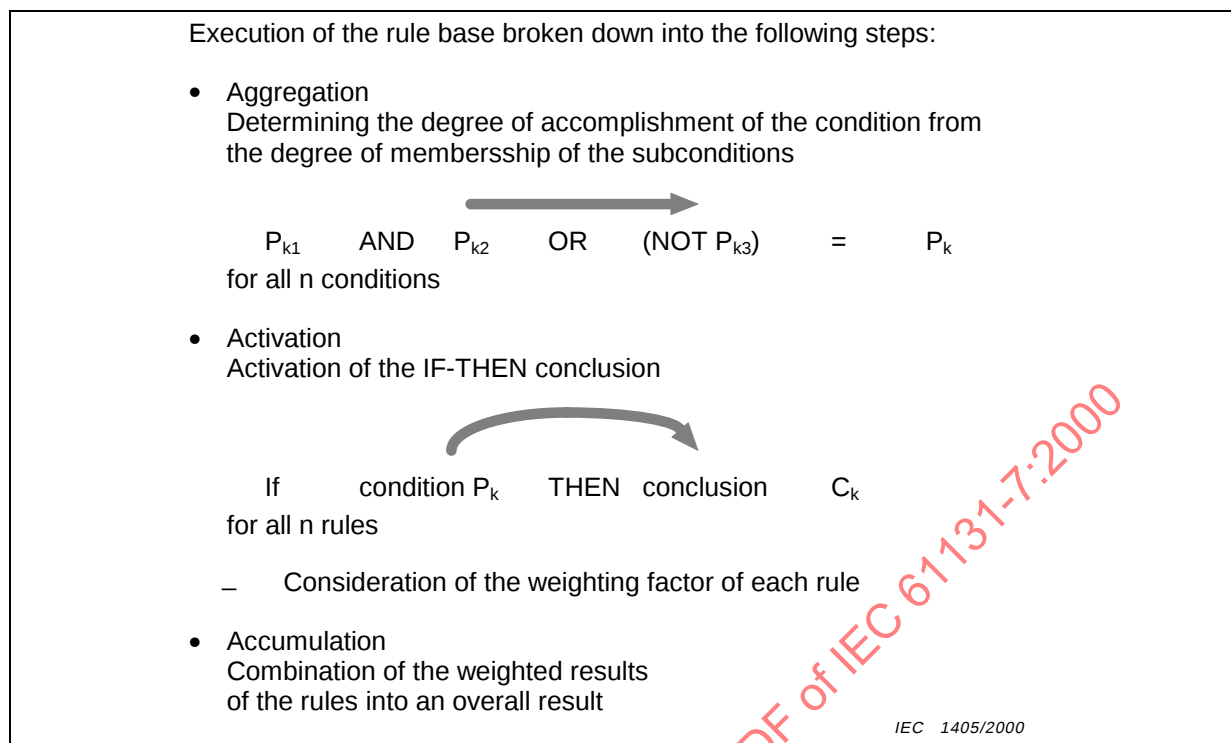


Figure A.9 – Elements of inference

– *Aggregation:*

If the condition consists of a single subcondition, then the validity of the *condition* is identical to that of the subcondition, i.e. the degree of accomplishment of the *condition* corresponds to the subcondition. However, if the *condition* consists of a combination of several subconditions, the degree of accomplishment must be determined by *aggregation* of the individual values. If an AND combination of subconditions exists, the degree of accomplishment is calculated by means of the AND *fuzzy operator*.

– *Activation:*

In the *conclusion*, *subconclusions* relate to the output variables. The *degree of membership* of the *conclusion* is then determined on the basis of the *degree of accomplishment* of the *condition* determined in aggregation (*Conclusion IF A THEN B*). In general, the MIN or PROD is used for activation.

If the rule base contains rules with weighting factors w_k , where $w_k \in [0, 1]$, this may be implemented by means of multiplication.

$$C_k^* = w_k \times C_k$$

(A.15)

– *Accumulation:*

The results of the *rules* are combined to obtain an overall result. The *maximum algorithm* is usually used for *accumulation*. Table A.1 shows the *operators* commonly used for the individual *inference* steps.

Depending on the combination of *operators* in the individual steps, different *inference* strategies are obtained. The best-known are the so-called *MaxMin Inference* and *MaxProd Inference*, which use the *maximum* for *accumulation* and the *minimum* or the *algebraic product* for *activation*. In the case of the *MaxMin Inference*, the *membership functions* of the *fuzzy sets* of the *conclusions* are limited to the *degree of accomplishment* of the *condition* and then, in turn, combined to create a *fuzzy set* by forming a maximum. In *MaxProd Inference*, in contrast, the *membership functions* of the *fuzzy sets* of the *conclusions* are weighted, i.e. multiplied, with the *degree of accomplishment* of the *condition* and then combined.

Tableau A.1 – Etapes d'inférence et algorithmes couramment utilisés

Etape d'inférence	Opérateurs	Algorithmes
<i>Agrégation</i>		
pour ET	<i>Minimum</i>	$a_k = \text{Min}\{a_{k1}(\times 1), a_{k2}(\times 2)\}$
pour OU	<i>Maximum</i>	$a_k = \text{Max}\{a_{k1}(\times 1), a_{k2}(\times 2)\}$
<i>Activation</i>		
conversion de la conclusion SI-ALORS		
	<i>Minimum</i>	$c_k' = \text{Min}\{a_k, \mu_{\alpha}(u)\}$
facteur de pondération de chaque règle		
	<i>Multiplication</i>	$c_k = \text{Mult}\{\omega_k, c_k'\} = \omega_k \times c_k'$
<i>Accumulation</i>	<i>Maximum</i>	$\mu = \text{MAX}\{c_i(u)\}$

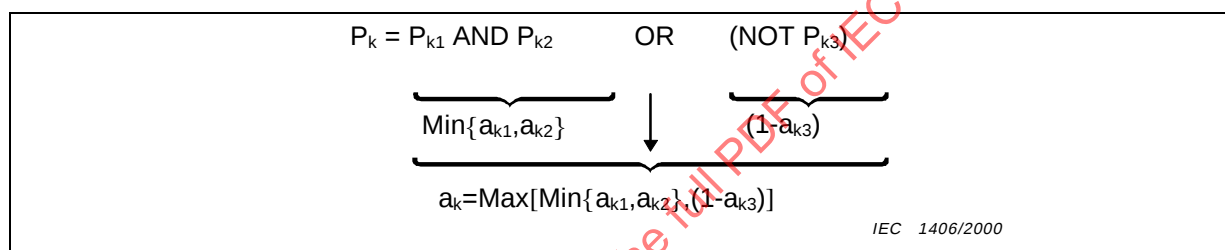


Figure A.10a – Exemple montrant les principes d'agrégation

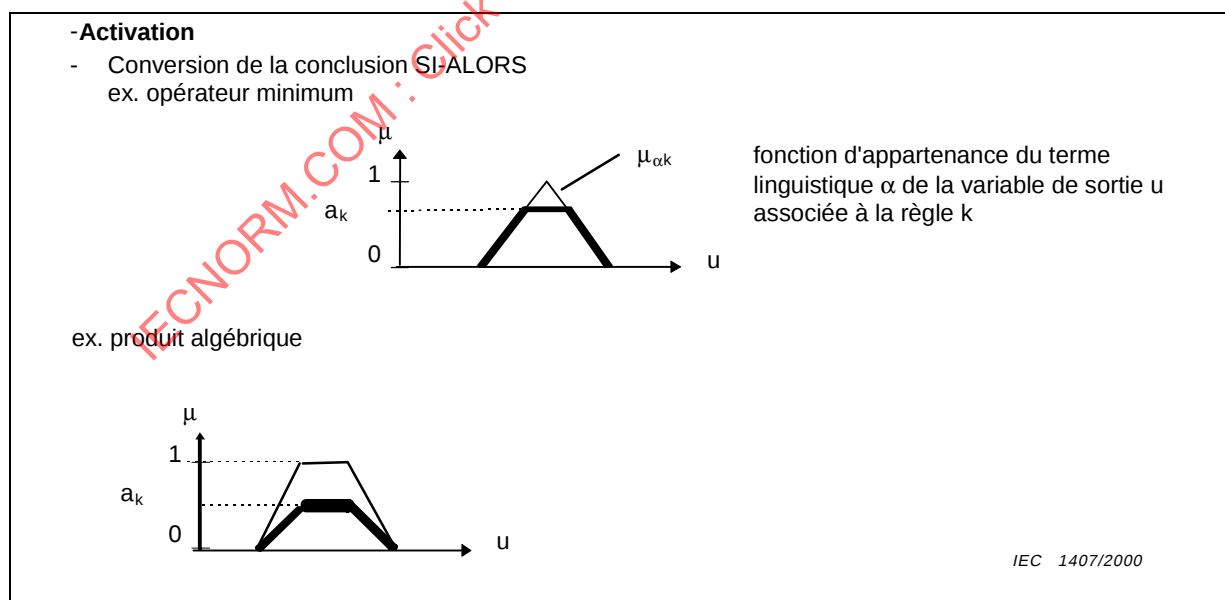


Figure A.10b – Principes d'activation (exemple)

Table A.1 – Inference steps and commonly used algorithms

Inference step	Operators	Algorithms
<i>Aggregation</i>		
for AND	<i>Minimum</i>	$a_k = \text{Min}\{a_{k1}(x1), a_{k2}(x2)\}$
for OR	<i>Maximum</i>	$a_k = \text{Max}\{a_{k1}(x1), a_{k2}(x2)\}$
<i>Activation</i>		
conversion of the IF-THEN-conclusion		
	<i>Minimum</i>	$c_k' = \text{Min}\{a_k, \mu_{\infty}(u)\}$
weighting factor of each rule		
	<i>Multiplication</i>	$c_k = \text{Mult}\{\omega_k, c_k'\} = \omega_k \times c_k'$
<i>Accumulation</i>	<i>Maximum</i>	$\mu = \text{MAX}\{c_i(u)\}$

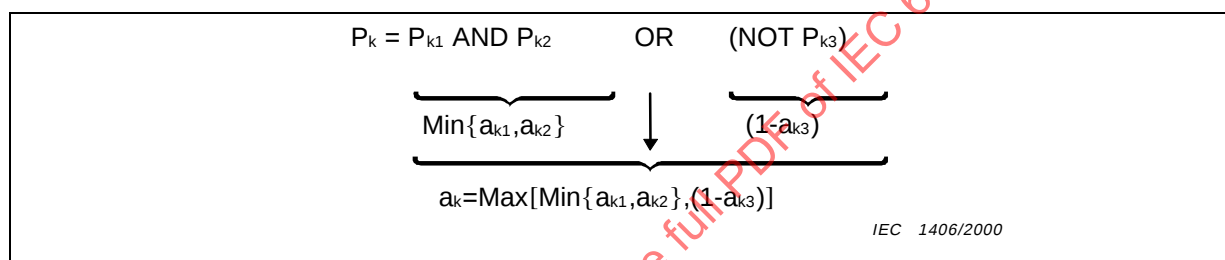


Figure A.10a – An example showing the principles of aggregation

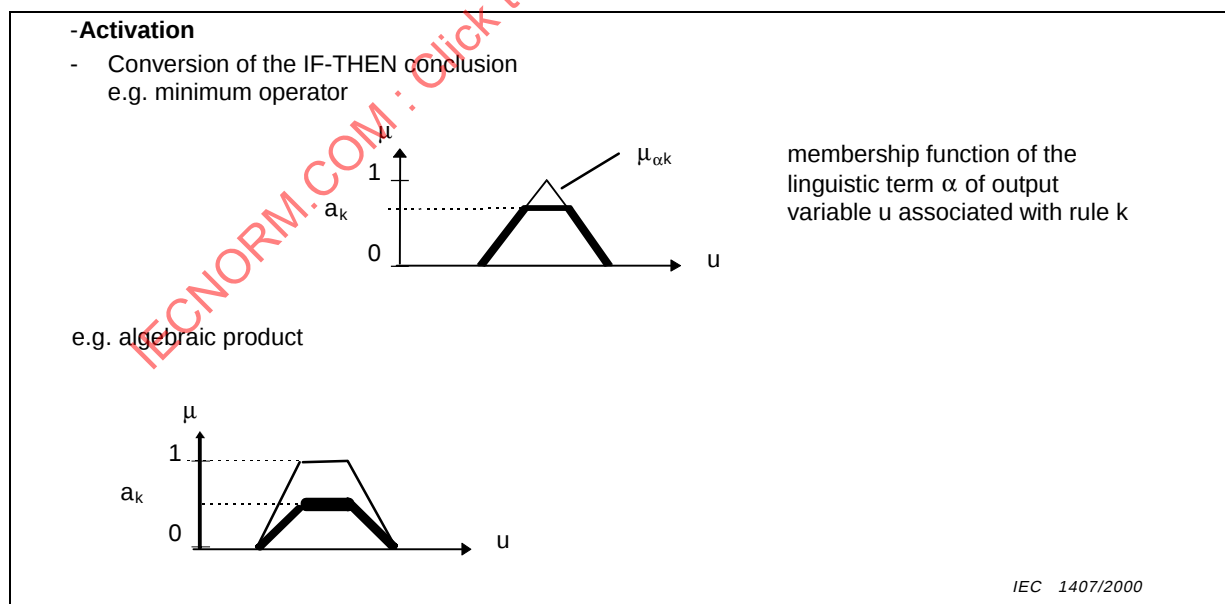


Figure A.10b – The principles of activation (as an example)

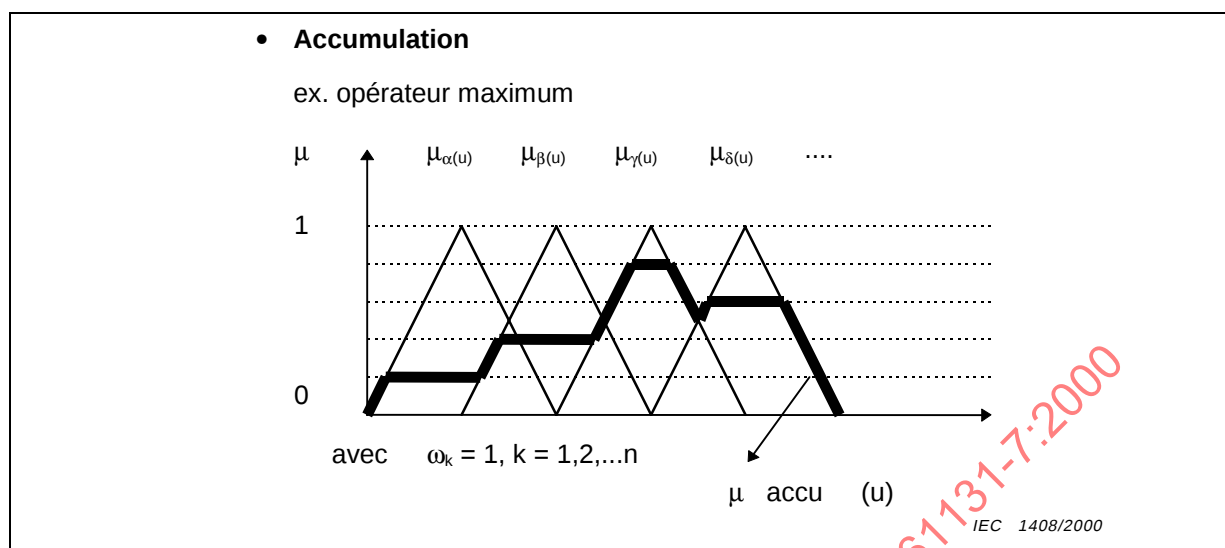


Figure A.10c – Principes d'accumulation (exemple)

A.2.4 Défuzzification (voir figure A.11)

Le résultat d'une *inférence* est un *ensemble flou* ou sa *fonction d'appartenance*. Ces informations floues ne peuvent pas être traitées directement par un système de contrôle; le résultat du processus d'*inférence* doit donc être converti en valeurs numériques «nettes». Dans ce contexte, le nombre «net» à déterminer (généralement un réel) devrait fournir une représentation adéquate des informations contenues dans l'*ensemble flou*.

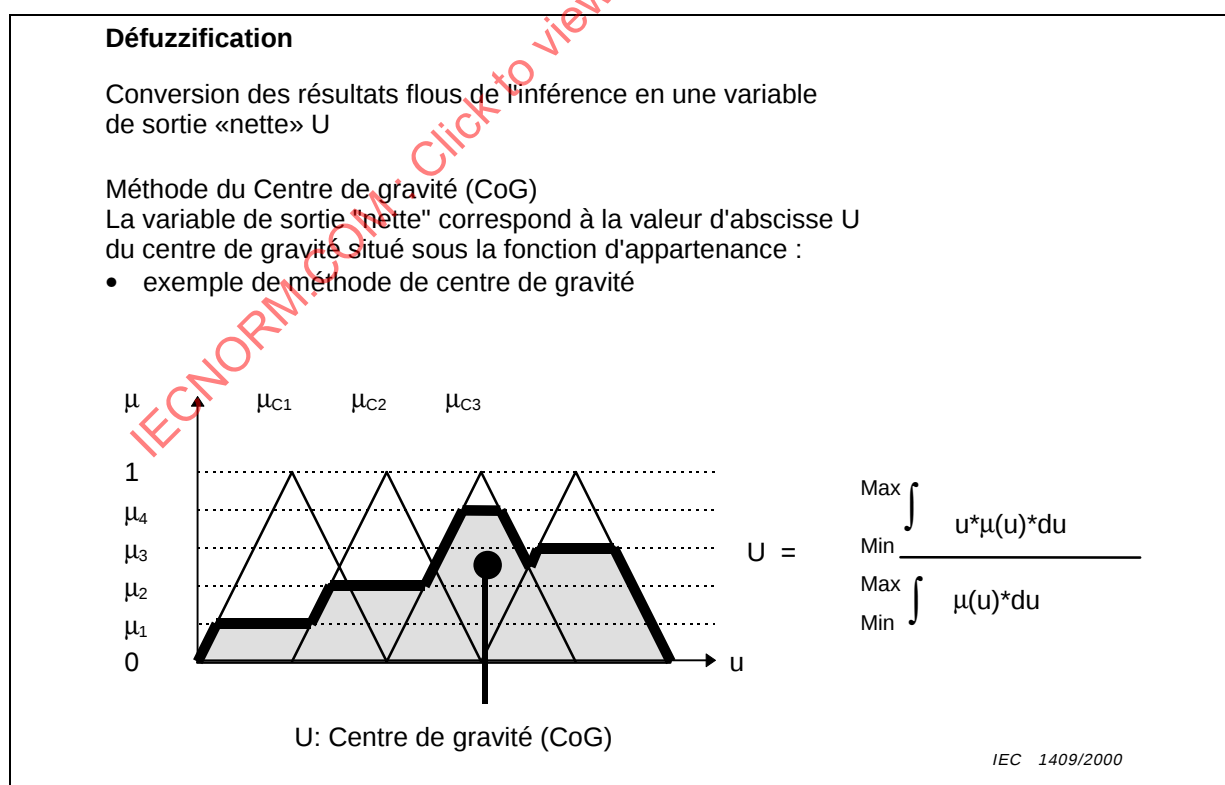


Figure A.11a – Méthodes de défuzzification

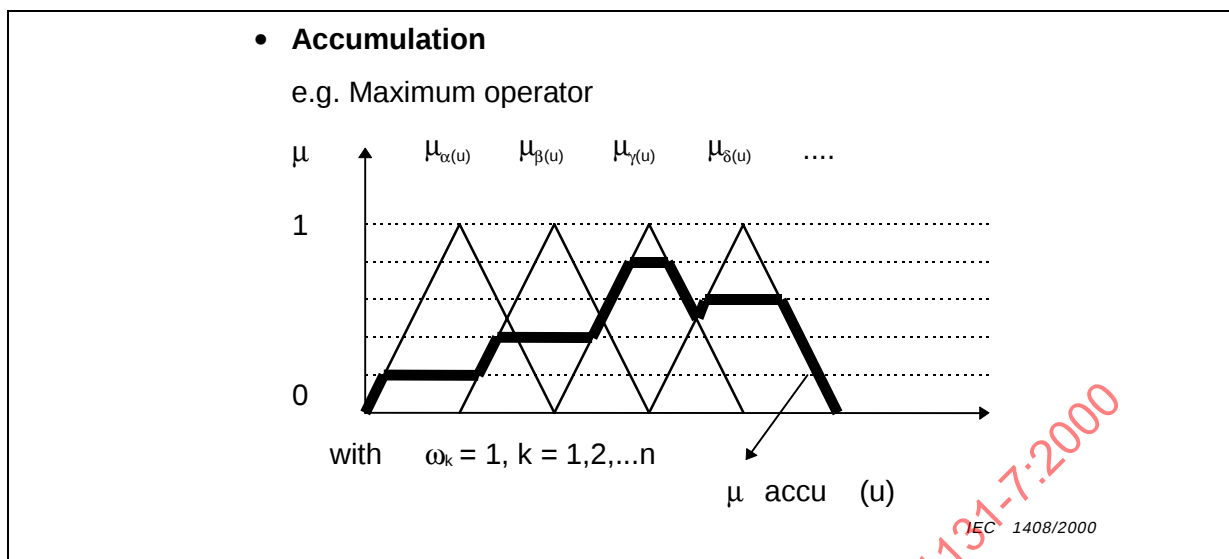


Figure A.10c – The principles of accumulation (as an example)

A.2.4 Defuzzification (see figure A.11)

Inference supplies a *fuzzy set* or its *membership function* as a result. A control element cannot directly process this fuzzy information; therefore, the result of the *inference* process has to be converted into *crisp* numerical values. In this context, the *crisp* number to be determined (generally a real number) should provide a good representation of the information contained in the *fuzzy set*.

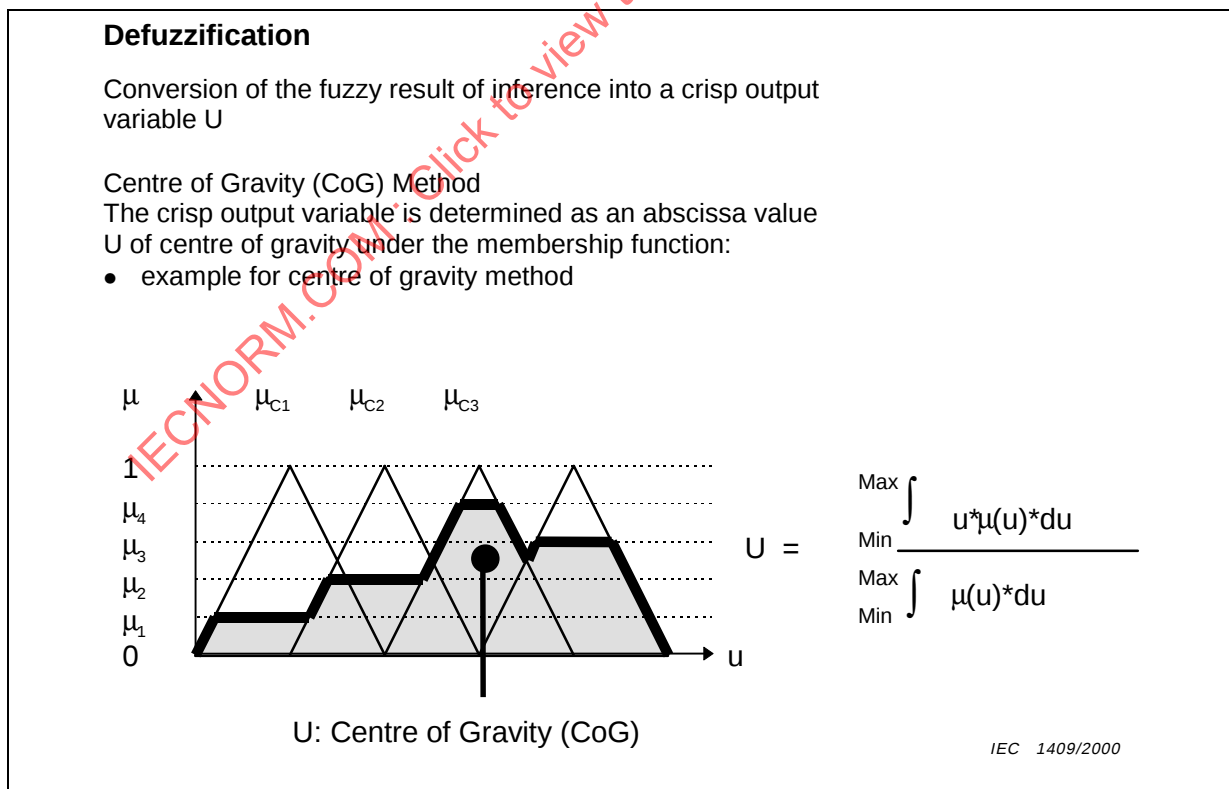


Figure A.11a – Methods of defuzzification

D'autres méthodes couramment utilisées sont les suivantes:

Maximum le plus à gauche (LM)

Calcul de la valeur de la variable de sortie pour laquelle la *fonction d'appartenance* de la sortie atteint son maximum situé le plus à gauche.

Maximum le plus à droite (RM)

Calcul de la valeur de la variable de sortie pour laquelle la *fonction d'appartenance* de la sortie atteint son maximum situé le plus à droite.

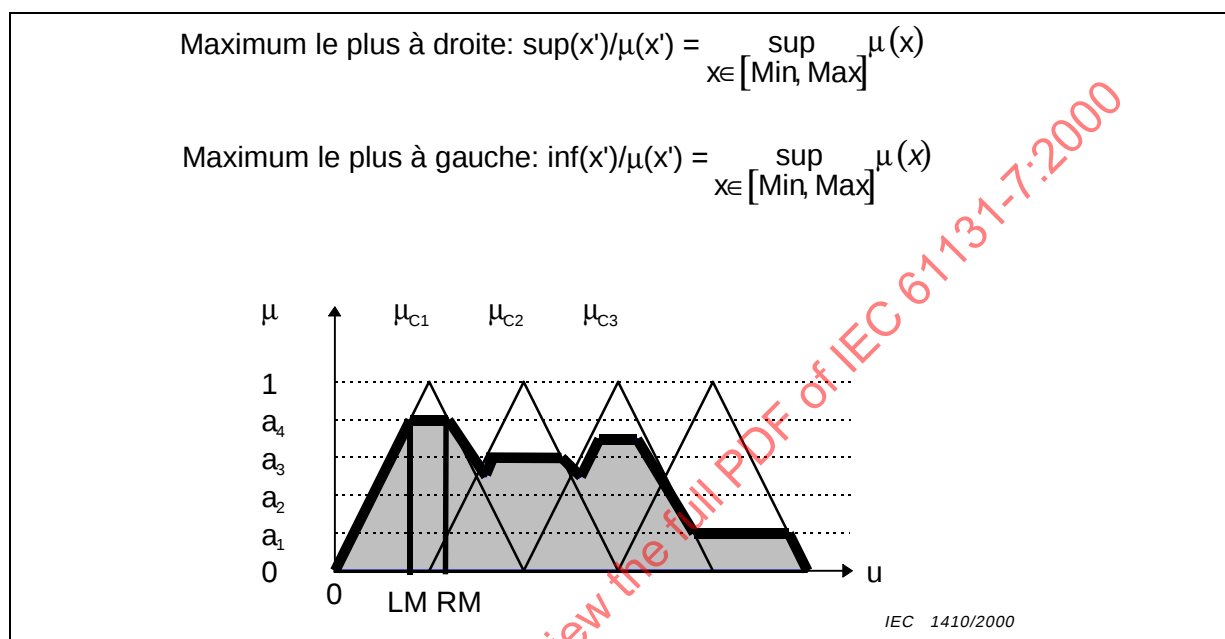


Figure A.11b – Comparaison entre maximum le plus à gauche et maximum le plus à droite

Méthode du Centre de surface

Ici, la valeur de sortie est déterminée par l'abscisse du segment vertical qui divise la région définie sous la *fonction d'appartenance* en deux régions de taille égale.

NOTE 1 Centre de gravité équivalent à Centroïde de surface

NOTE 2 Centre de surface équivalent à Bissectrice de surface

NOTE 3 CoA ne s'applique pas si des singletons sont utilisés.

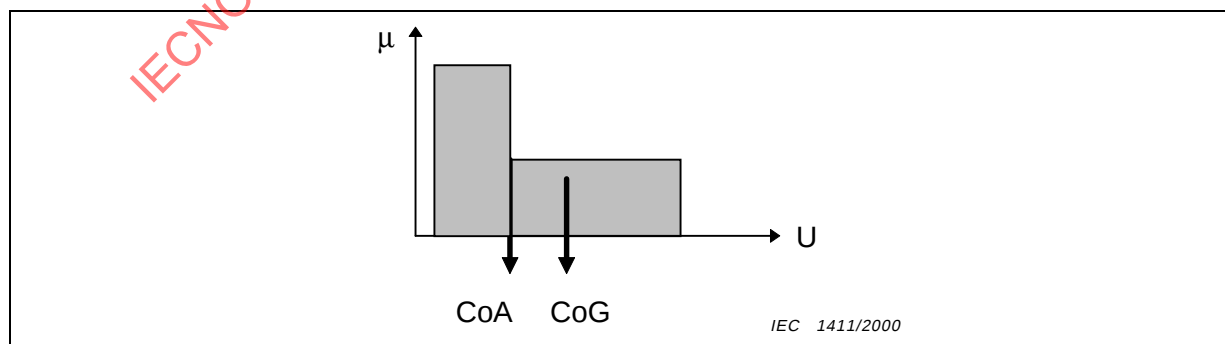


Figure A.11c – Comparaison entre Centre de surface et Centre de gravité

NOTE Les formules correspondant aux méthodes de défuzzification CoA et CoG sont présentées au tableau 2

Further common methods are as follows:

Left Most Maximum LM

The value of the output variable is determined for which the *membership function* of the output reaches its leftmost maximum.

Right Most Maximum RM

The value of the output variable is determined for which the *membership function* of the output reaches its rightmost maximum.

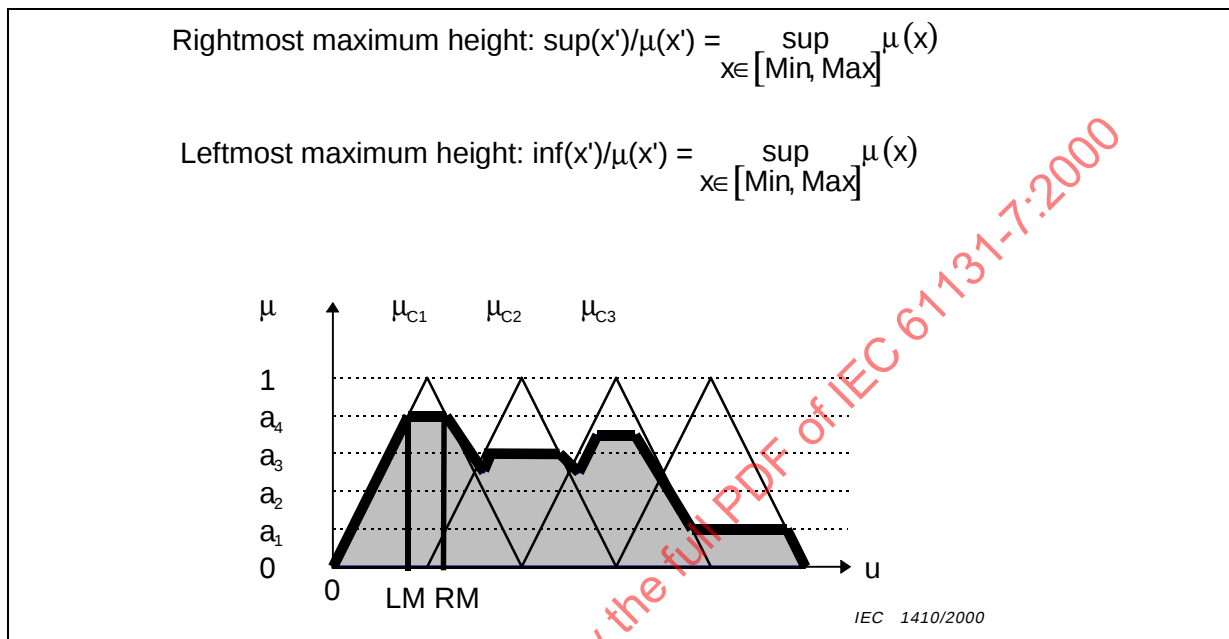


Figure A.11b – Difference between Left Most Maximum and Right Most Maximum

Centre of Area Method

Here, the output value is determined as the abscissa value of the centre of which divides the area under the *membership function* into 2 areas of equal size.

NOTE 1 Centre of Gravity is equivalent to Centroid of Area.

NOTE 2 Centre of Area is equivalent to Bisector of Area.

NOTE 3 CoA is not applicable if singletons are used.

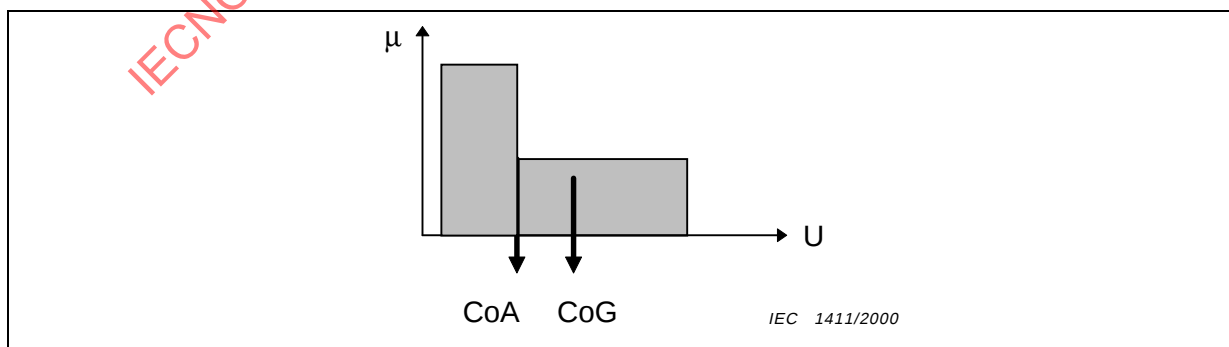


Figure A.11c – Difference between Centre of Area and Centre of Gravity

NOTE Formulae of the Defuzzification methods CoA and CoG are given in table 2.

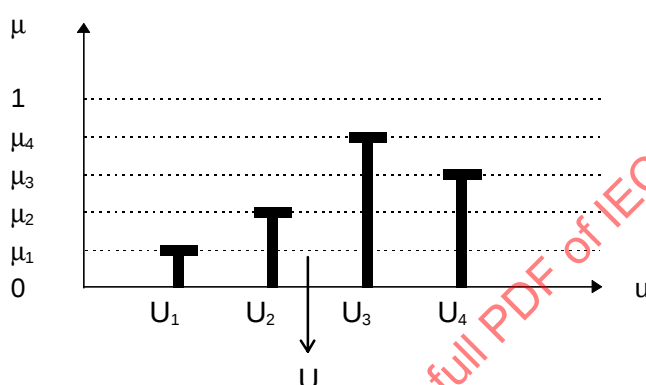
Défuzzification

Lorsque les fonctions d'appartenance des valeurs de sortie sont des singletons, le calcul est effectué par :

- Méthode du centre de gravité de singleton (COGS)

$$U = \frac{\sum_{i=1}^p [U_i * \mu_i]}{\sum_{i=1}^p [\mu_i]}$$

où μ représente les résultats d'accumulation



Dans le cas de singletons, le centre de surface ne s'applique pas. Exemple pour les 4 singletons ci-dessus :

$$U = \frac{[U_1 * \mu_1 + U_2 * \mu_2 + U_3 * \mu_3 + U_4 * \mu_4]}{[\mu_1 + \mu_2 + \mu_3 + \mu_4]}$$

IEC 1412/2000

Figure A.11d – Méthodes de défuzzification

A.3 Performances du contrôle flou

Du point de vue des technologies de l'information, le contrôle flou est un système expert à base de règles. Du point de vue des technologies des systèmes de contrôle, c'est un automate à caractéristique généralement non linéaire. La figure A.12 illustre des exemples de courbes caractéristiques de contrôle flou. Les valeurs en cours de ses variables de sortie dépendent exclusivement des valeurs en cours des variables d'entrée, et non pas de valeurs précédentes, excepté dans le cas où aucune règle n'est active et qu'aucune valeur par défaut n'a été définie. Si l'automate est pourvu de fonctions dynamiques, il faut que celles-ci soient externes au bloc fonction flou.

Ces fonctions dynamiques sont généralement des différentielles et intégrales du premier degré. Les variables de sortie de ces fonctions constituent des variables d'entrée supplémentaires pour le contrôle flou. Cela s'applique également à l'erreur de contrôle, dont la formation est extérieure au contrôle flou. Réciproquement, les variables de sortie de contrôle flou peuvent être transmises à des opérateurs pour être traitées et corrigées (un élément d'intégration pour des algorithmes de vitesse, par exemple) ou réparties entre différents éléments de contrôle.

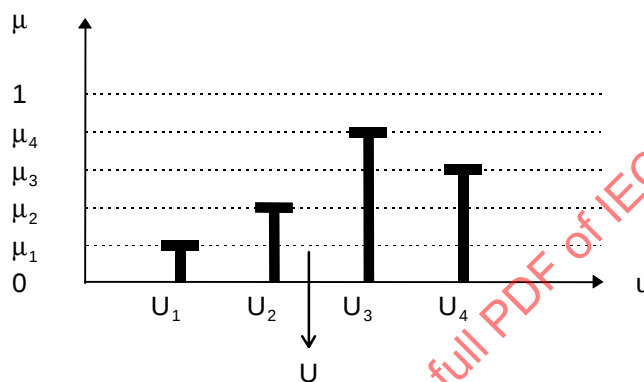
Defuzzification

If the membership functions of the output values are singletons, the calculation is given by:

- Centre of Gravity Method for Singleton (COGS)

$$U = \frac{\sum_{i=1}^p [U_i * \mu_i]}{\sum_{i=1}^p [\mu_i]}$$

where μ are the results of accumulation



In the case of singletons, the centre of area is not applicable.
Example for the 4 singletons above:

$$U = \frac{[U_1 * \mu_1 + U_2 * \mu_2 + U_3 * \mu_4 + U_4 * \mu_3]}{[\mu_1 + \mu_2 + \mu_3 + \mu_4]}$$

IEC 1412/2000

Figure A.11d – Methods of defuzzification

A.3 Performance of Fuzzy control

From a point of view of information technology, fuzzy control is a rule-based expert system. From the point of view of control systems technology, it is a generally non-linear characteristic field controller. Figure A.12 shows examples of fuzzy control characteristic curves. The current values of its output variables depend exclusively on the current values of the input variables, and not on previous values, except in the case where no rule is active and no default value has been defined. If the controller should be realised with dynamic behaviour, the dynamic functions must be provided external to the fuzzy function block.

These are usually differentiating and integrating elements of the first order. The output variables of these functions are additional input variables for fuzzy control. This also applies to the control error, which likewise has to be formed outside fuzzy control. Conversely, the output variables of fuzzy control may be passed to operators for processing of the correcting variables, for example an integration element for speed algorithms, or distributed among different control elements.

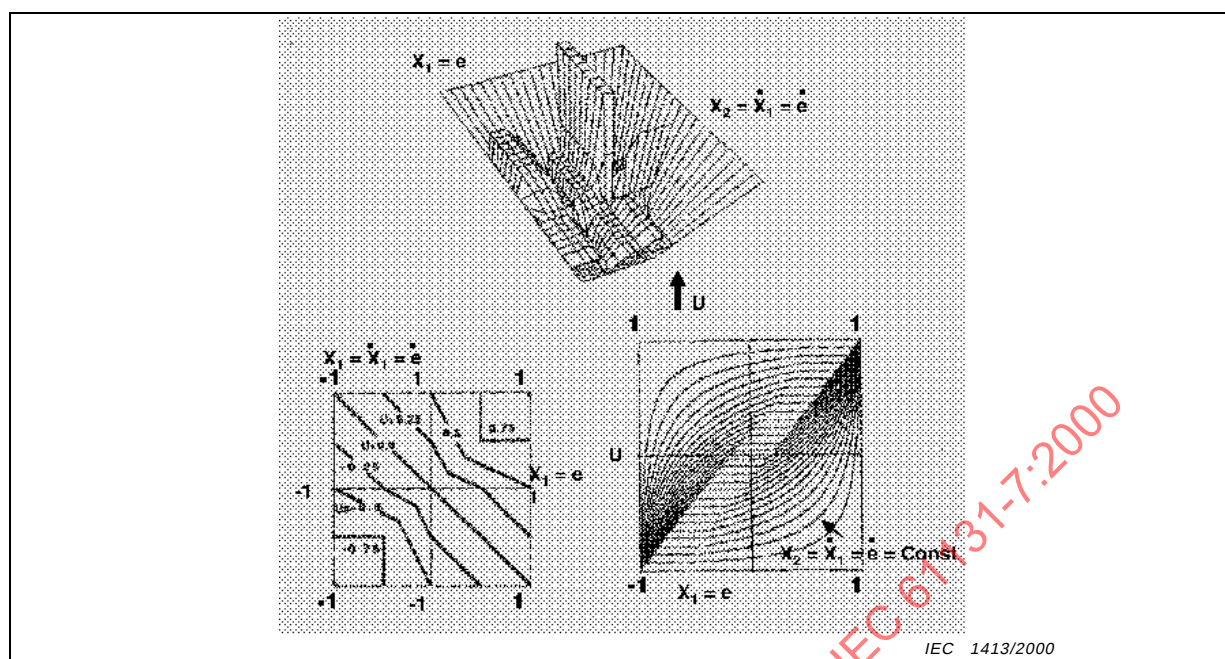


Figure A.12 – Exemples de courbes de caractéristiques de contrôle flow

La structure fondamentale d'un automate à logique floue est représentée à la figure A.13a; un exemple d'automate est illustré par la figure A.13b: l'erreur de contrôle (e) résulte de la différence entre le point de consigne et la variable de procédé. Cette différence, ainsi que sa dérivée en fonction du temps, sont transmises au contrôle flow sous la forme de deux variables d'entrée qui sont indépendantes de sa perspective. La variable de correction est déduite de la variable de sortie par intégration dans le temps. Si la conception de la base de règles permet de décrire un algorithme de vitesse, l'automate à logique floue présente un comportement dynamique semblable à celui d'un automate PI (intégral proportionnel).

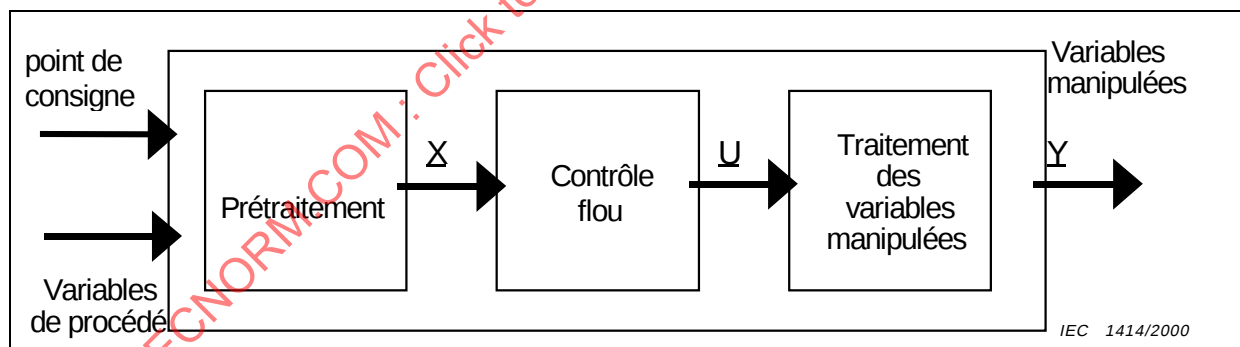


Figure A.13a – Automate à base de logique floue: structure fondamentale

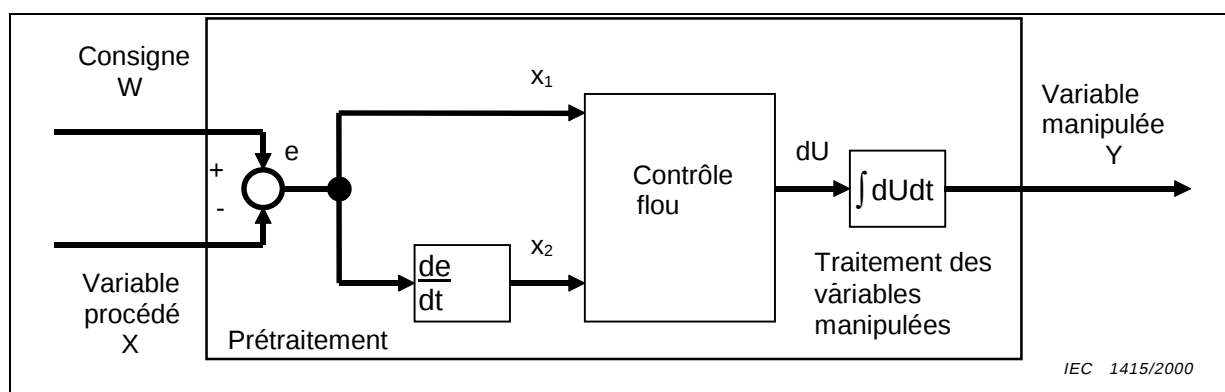


Figure A.13b – Exemple d'automate à base de logique floue

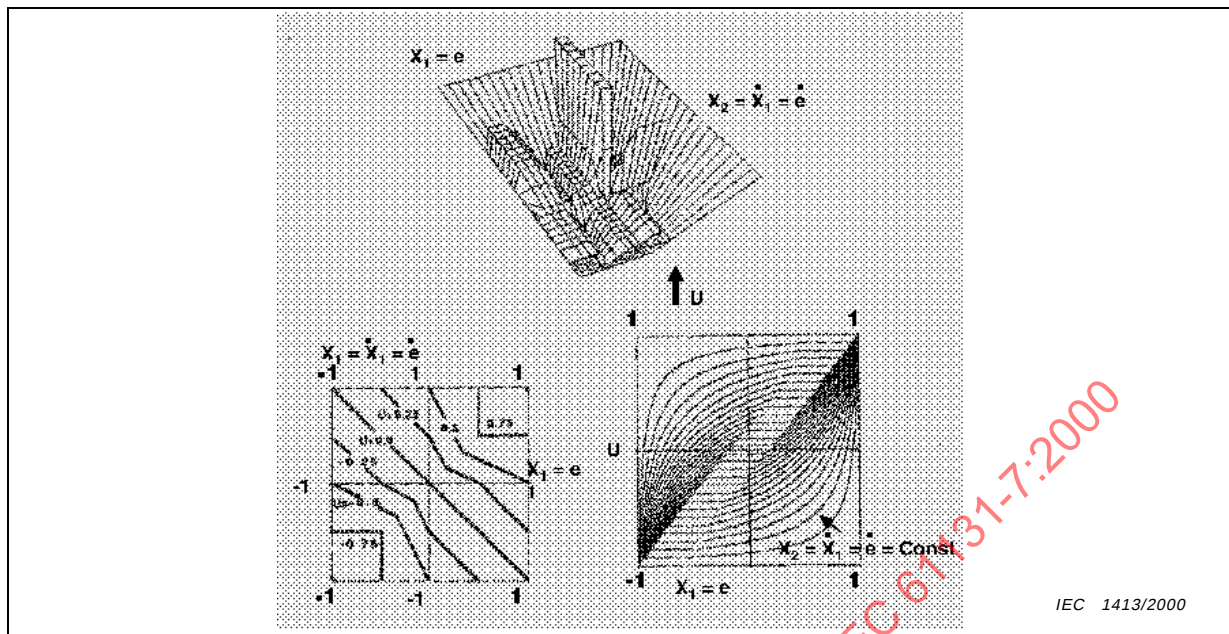


Figure A.12 – Examples of fuzzy control characteristic curves

The fundamental structure of a fuzzy-based controller is shown in figure A.13a, while figure A.13b gives an example: the control error (e) is formed from the difference between the setpoint and the process variable. This difference, as well as its derivative with respect to time, is transmitted to fuzzy control as two input variables which are independent from its perspective. The correcting variable is derived from the output variable by means of integration over time. If the rule base is designed in such a way that a velocity algorithm is described, the fuzzy-based controller thus displays dynamic behaviour similar to PI.

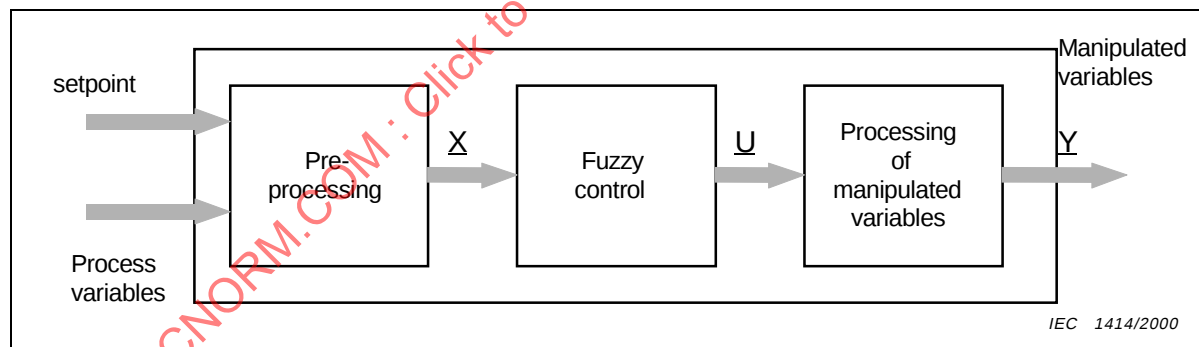


Figure A.13a – Fuzzy-based controller: Fundamental structure

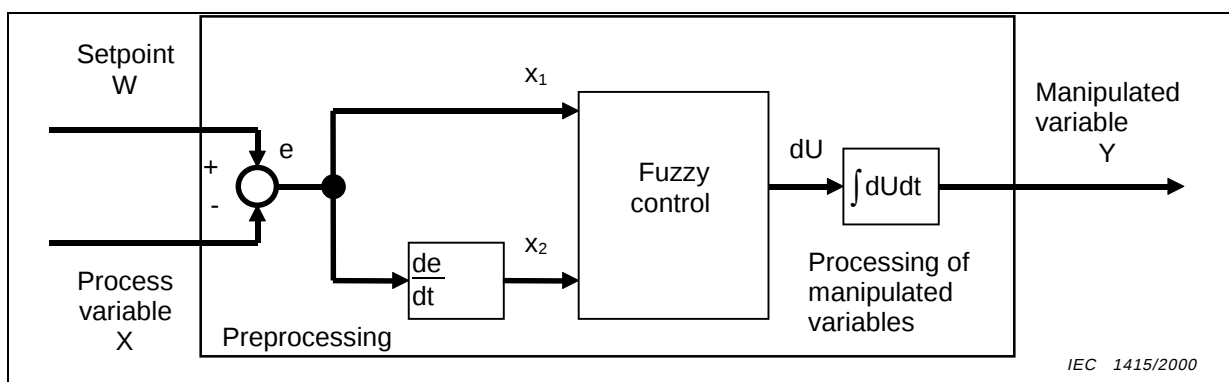


Figure A.13b – Example of a Fuzzy-based controller

Annexe B (informative)

Exemples

Dans un de leurs principaux domaines d'application, les automates programmables sont combinés à des automates PID conventionnels, l'objectif étant d'améliorer la qualité de contrôle de ce type d'automate. Les exemples suivants en illustrent les principales possibilités, afin de donner une idée des situations dans lesquelles le contrôle flou peut intervenir.

B.1 Précontrôle

L'automate flou complète l'automate à boucle fermée conventionnel par un signal de correction pour la valeur manipulée (voir figure B.1).

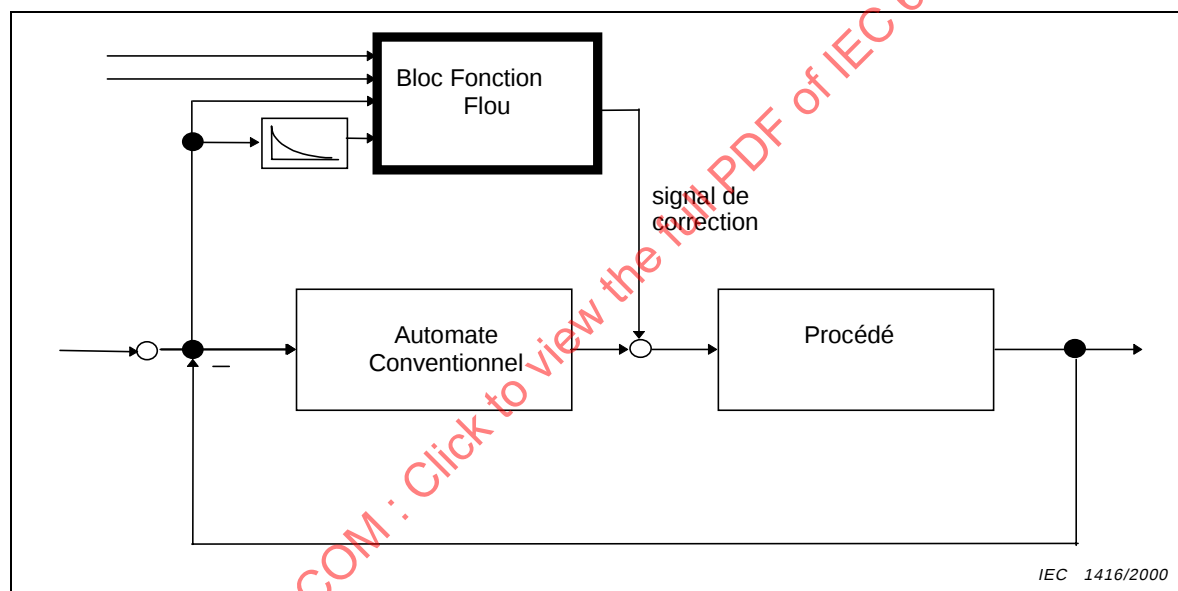


Figure B.1 – Exemple de précontrôle

Annex B (informative)

Examples

One of the main application areas in the world of Programmable Controllers is in combination with conventional PID-controllers in order to improve the control quality of the PID controller. The following examples show the principal possibilities in general to give an idea where fuzzy control may be used.

B.1 Pre-control

The fuzzy controller supplements the conventional closed loop controller by a corrective signal for the manipulated value (see figure B.1).

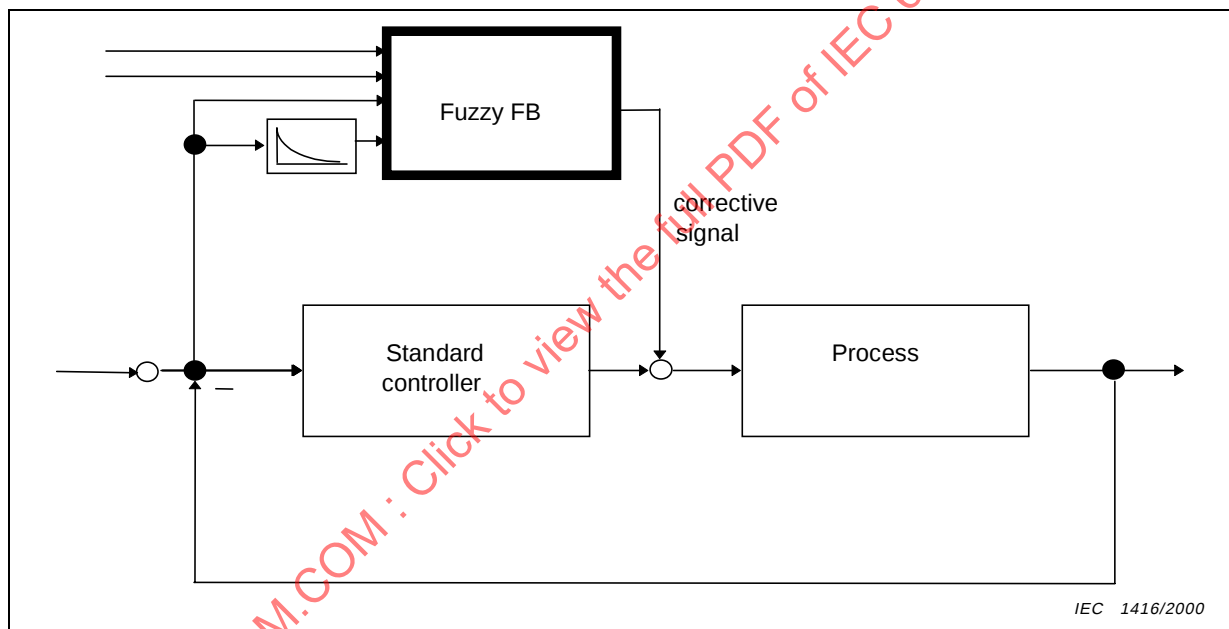


Figure B.1 – Example of a pre-control

B.2 Adaptation des paramètres d'automate PID conventionnel

L'automate flou permet d'adapter les paramètres de contrôle d'un automate PID (voir figure B.2).

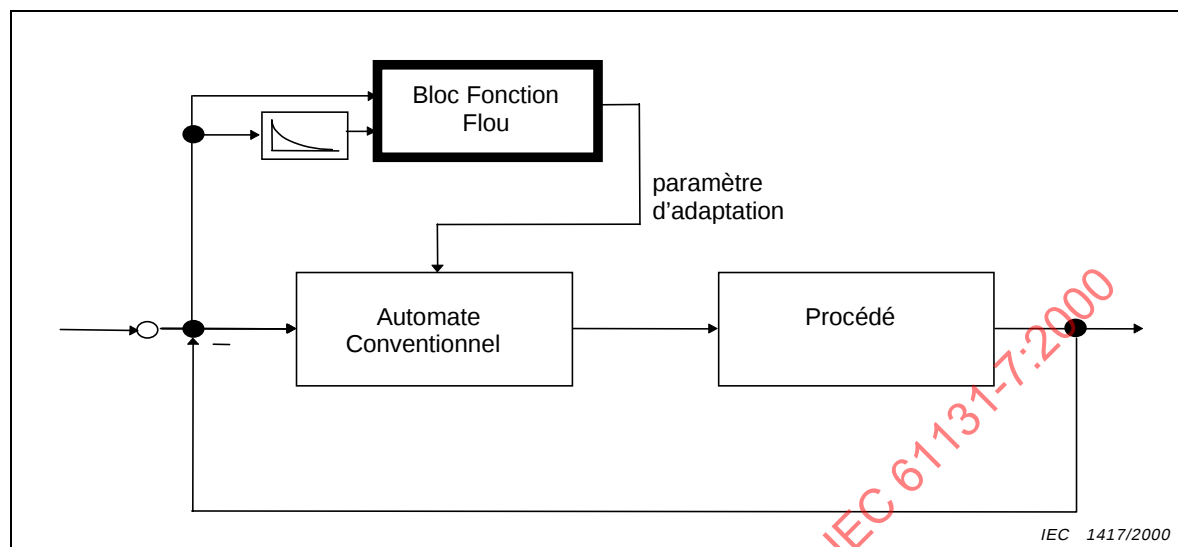


Figure B.2 – Exemple d'adaptation d'un paramètre

B.3 Contrôle flou direct d'un procédé

Un autre domaine d'application concerne l'incorporation directe de connaissances empiriques et de stratégies de contrôle linguistique dans les automatismes industriels. Cela s'applique à de nombreux procédés dans lesquels l'intervention d'un opérateur humain serait nécessaire (voir figure B.3).

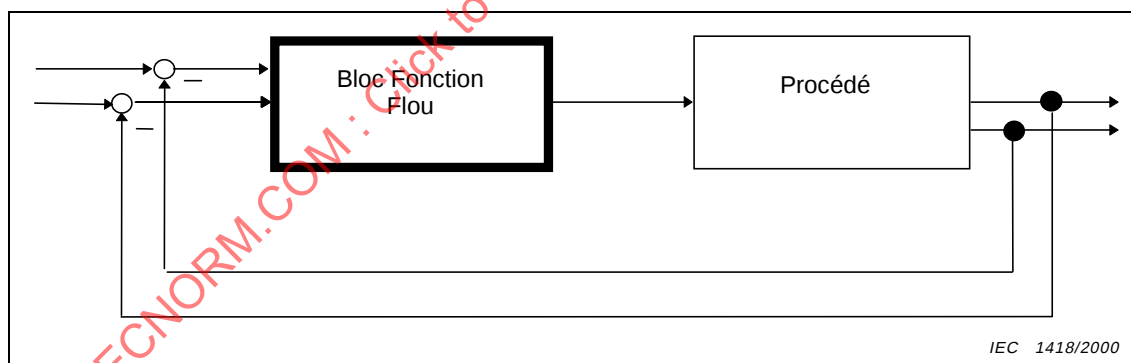


Figure B.3 – Exemple de contrôle flou direct

B.2 Parameter adaptation of conventional PID controllers

The fuzzy controller is used to adapt the control parameters of a PID controller (see figure B.2).

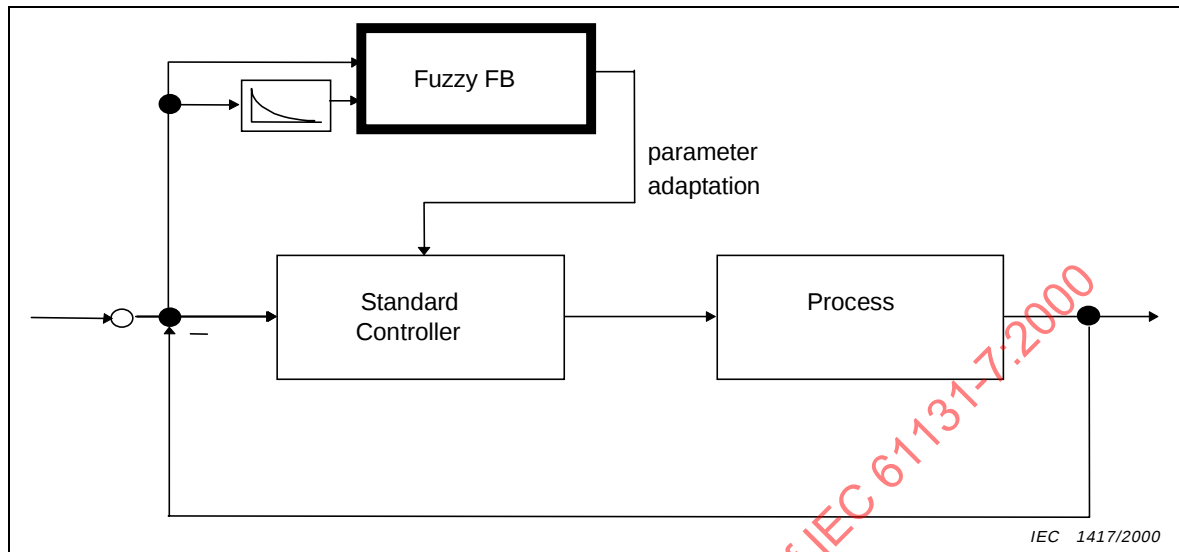


Figure B.2 – Example of a parameter adaptation

B.3 Direct fuzzy control of a process

Another area of application is to include empirical process knowledge and linguistic control strategies directly into industrial automation. This applies to many processes where operator intervention would be necessary (see figure B.3).

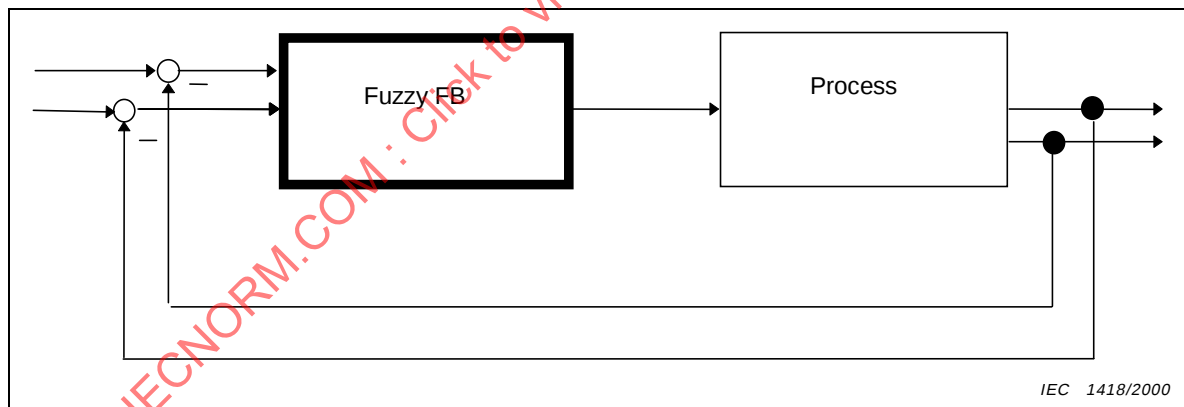


Figure B.3– Example of a direct fuzzy control

Annexe C (informative)

Exemple d'application industrielle – Grue à conteneurs

Les grues à conteneurs sont utilisées dans la plupart des ports pour charger et décharger les conteneurs transportés par navire. Ces conteneurs sont soulevés un par un par des câbles fixés sur la tête de la grue, qui ensuite se déplace le long d'une poutre horizontale. Au démarrage de ce mouvement horizontal, le conteneur suspendu se met à osciller, comme illustré à la figure C.1. Bien que ce balancement ne présente aucun problème particulier pendant le déplacement, un conteneur qui oscille est impossible à détacher.

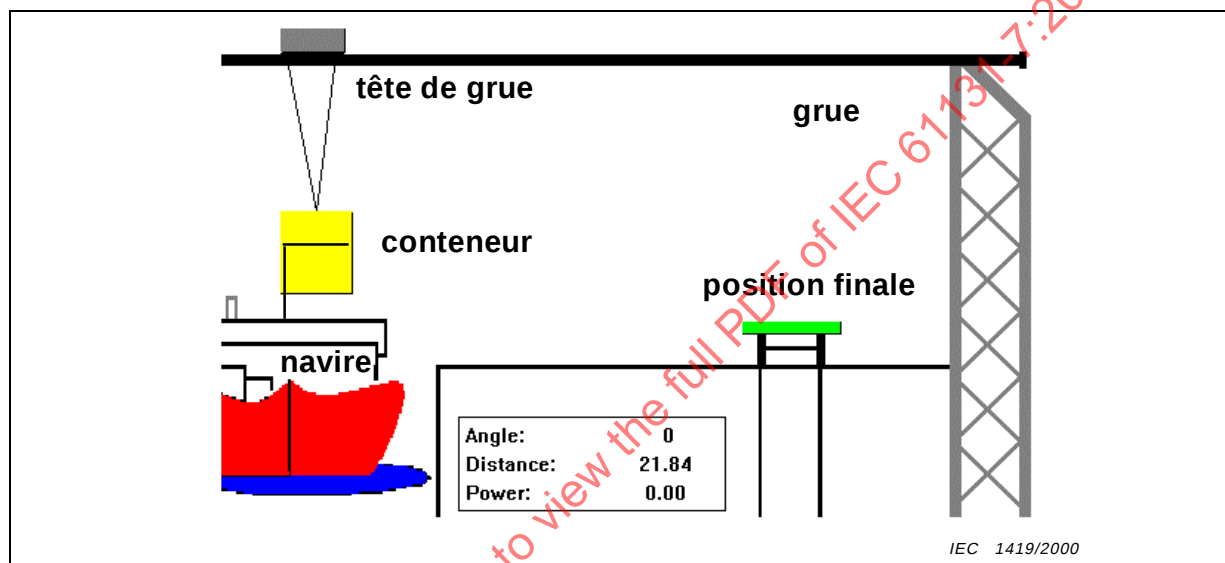


Figure C.1 – Exemple d'application industrielle – Grue à conteneurs

L'analyse des actions de l'opérateur révèle que ce dernier fait appel à des méthodes empiriques pour définir sa stratégie de contrôle.

- a) Démarrer à puissance modérée.
- b) Si, après démarrage, la distance à parcourir reste importante, ajuster la puissance de sorte que le conteneur se retrouve légèrement derrière la tête de la grue.
- c) Si la position finale est proche, réduire la puissance pour que le conteneur devance légèrement la tête de la grue.
- d) Lorsque le conteneur est près de sa position finale, utiliser +/- la puissance moyenne en fonction du sens de l'oscillation.
- e) Lorsque le conteneur est au-dessus de sa position finale et que son oscillation est nulle, arrêter le moteur.

Pour automatiser le contrôle de la grue, on utilise des capteurs qui mesurent la position de la tête de grue («Distance») et l'angle d'oscillation du conteneur («Angle»). La sortie détermine la puissance du moteur. Des variables linguistiques doivent d'abord être définies pour chaque variable. Les variables linguistiques de distance, angle et puissance moteur (voir figures C.2, C.3 et C.4) sont représentées par cinq termes linguistiques. Les fonctions d'appartenance ont des formes de rampes, de triangles et de singletons. Les variables linguistiques et termes linguistiques utilisés sont représentés sur les figures suivantes.

Annex C (informative)

Industrial example – Container crane

Container cranes are used to load and unload containers to and from ships in most harbours. They pick up single containers with cables that are mounted at the crane head. The crane head moves on a horizontal track. When a container is picked up and the crane head starts to move, the container begins to sway as shown in figure C.1. While sway is no problem during transportation, a swaying container cannot be released.

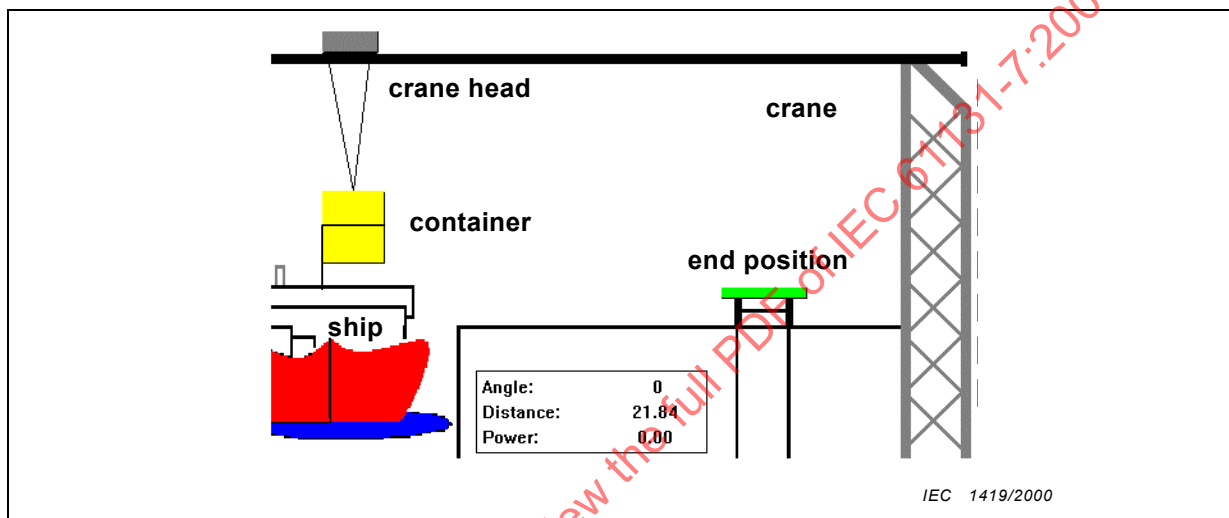


Figure C.1 – Industrial example – Container crane

The analysis of the operator's actions reveals that the operator uses some "rules of thumb" to describe his control strategy.

- a) Start with medium power.
- b) If you got started and you are still far away from target, adjust the motor power so that the container gets a little behind the crane head.
- c) If you are closer to the target, reduce power so the container gets a little ahead of the crane head.
- d) When the container is close to target position, use +/- medium power, depending upon sway direction.
- e) When the container is over the target and the sway is zero, stop the motor.

To automate the control of this crane, sensors for the crane head position ("Distance") and the angle of the container sway ("Angle") are employed. The output is the motor power. First, linguistic variables have to be defined for all variables. The linguistic variables distance, angle and motor power (see figures C.2, C.3 and C.4) are divided into five linguistic terms. The used forms of the membership function are ramps, triangles and singletons. The following figures show the linguistic variables and the linguistic terms.

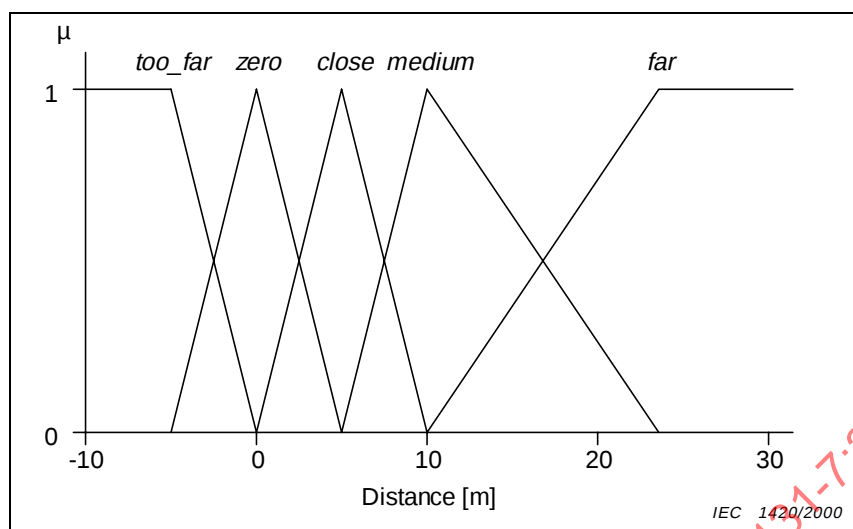


Figure C.2 – Variable linguistique «Distance» entre tête de grue et position finale

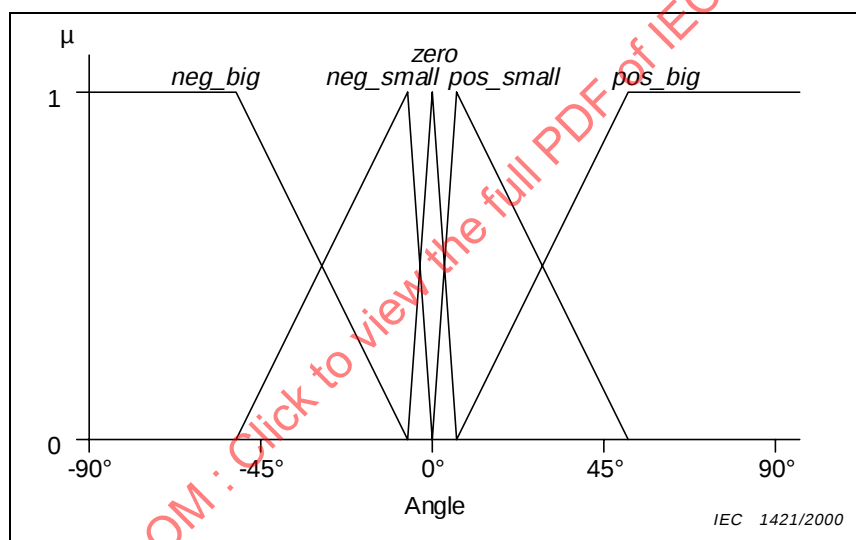


Figure C.3 – Variable linguistique «Angle» entre conteneur et tête de grue

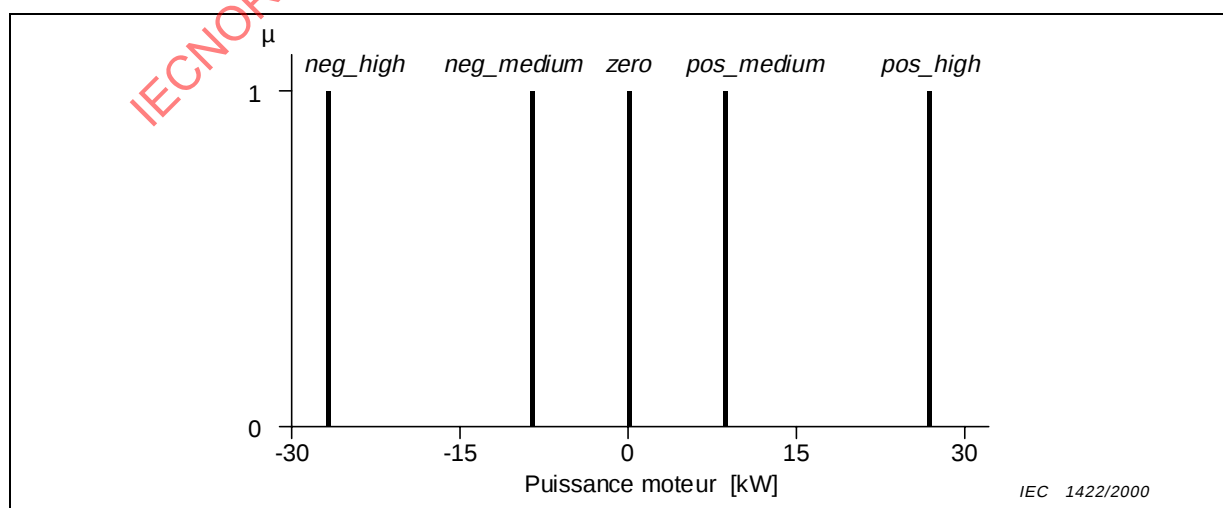


Figure C.4 – Variable linguistique «Puissance»