

INTERNATIONAL
STANDARD

ISO/IEC
2382-7

NORME
INTERNATIONALE

Second edition
Deuxième édition
1989-10-15

Information technology — Vocabulary —

Part 07:
Computer programming

Technologies de l'information — Vocabulaire —

Partie 07:
Programmation des ordinateurs



Reference number
Numéro de référence
ISO/IEC 2382-7 : 1989 (E/F)

Contents

	Page
Introduction	vi
Section 1: General	
1.1 Scope	1
1.2 Normative references	1
1.3 Principles and rules followed	2
1.3.1 Definition of an entry	2
1.3.2 Organization of an entry	2
1.3.3 Classification of entries	2
1.3.4 Selection of terms and wording of definitions	3
1.3.5 Multiple meanings	3
1.3.6 Abbreviations	3
1.3.7 Use of parentheses	3
1.3.8 Use of brackets	3
1.3.9 Use of terms printed in italic typeface in definitions and the use of an asterisk	4
1.3.10 Spelling	4
1.3.11 Organization of the alphabetical index	4
Section 2: Terms and definitions	
07 Computer programming	5
07.01 General terms	5
07.02 Programming concepts	6
07.03 Program preparation	9
07.04 Program execution	13
07.05 Program debugging and verification	14
07.06 Instructions and addresses	16
Alphabetical indexes	
English	19
French	22

© ISO/IEC 1989

All rights reserved. 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./Droits de reproduction réservés. 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.

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland/Imprimé en Suisse

Sommaire

	Page
Introduction	vii
Section 1: Généralités	
1.1 Domaine d'application	1
1.2 Références normatives	1
1.3 Principes d'établissement et règles suivies	2
1.3.1 Définition de l'article	2
1.3.2 Constitution d'un article	2
1.3.3 Classification des articles	2
1.3.4 Choix des termes et des définitions	3
1.3.5 Pluralité de sens ou polysémie	3
1.3.6 Abréviations	3
1.3.7 Emploi des parenthèses	3
1.3.8 Emploi des crochets	3
1.3.9 Emploi dans les définitions de termes imprimés en caractères italiques et de l'astérisque	4
1.3.10 Mode d'écriture et orthographe	4
1.3.11 Constitution de l'index alphabétique	4
Section 2: Termes et définitions	
07 Programmation des ordinateurs	5
07.01 Termes généraux	5
07.02 Notions liées à la programmation	6
07.03 Préparation des programmes	9
07.04 Exécution de programmes	13
07.05 Mise au point et vérification des programmes	14
07.06 Instructions et adresses	16
Index alphabétiques	
Anglais	19
Français	22

Foreword

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

In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1. Draft International Standards adopted by the joint technical committee are circulated to national bodies for approval before their acceptance as International Standards. They are approved in accordance with procedures requiring at least 75 % approval by the national bodies voting.

International Standard ISO/IEC 2382-7 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*.

This second edition cancels and replaces the first edition (ISO 2382-7 : 1977), of which it constitutes a technical revision.

ISO/IEC 2382 will consist of some twenty-eight parts, under the general title *Information technology — Vocabulary*.

Avant-propos

L'ISO (Organisation internationale de normalisation) et la CEI (Commission électro-technique internationale) forment ensemble un système consacré à la normalisation internationale considérée comme un tout. Les organismes nationaux membres de l'ISO ou de la CEI participent au développement de Normes internationales par l'intermédiaire des comités techniques créés par l'organisation concernée afin de s'occuper des différents domaines particuliers de l'activité technique. Les comités techniques de l'ISO et de la CEI collaborent dans des domaines d'intérêt commun. D'autres organisations internationales, gouvernementales ou non gouvernementales, en liaison avec l'ISO et la CEI participent également aux travaux.

Dans le domaine des technologies de l'information, l'ISO et la CEI ont créé un comité technique mixte, l'ISO/CEI JTC 1. Les projets de Normes internationales adoptés par le comité technique mixte sont soumis aux organismes nationaux pour approbation, avant leur acceptation comme Normes internationales. Les Normes internationales sont approuvées conformément aux procédures qui requièrent l'approbation de 75 % au moins des organismes nationaux votants.

La Norme internationale ISO/CEI 2382-7 a été élaborée par le comité technique mixte ISO/CEI JTC 1, *Technologies de l'information*.

Cette deuxième édition annule et remplace la première édition (ISO 2382-7 : 1977), dont elle constitue une révision technique.

ISO/CEI 2382 comprendra une trentaine de parties, présentées sous le titre général *Technologies de l'information — Vocabulaire*.

Introduction

Information processing gives rise to numerous international exchanges of both an intellectual and a material nature. These exchanges often become difficult, either because of the great variety of terms used in various fields or languages to express the same concept, or because of the absence or imprecision of the definitions of useful concepts.

To avoid misunderstandings and to facilitate such exchanges it is essential to clarify the concepts, to select terms to be used in various languages or in various countries to express the same concept, and to establish definitions providing satisfactory equivalents for the various terms in different languages.

ISO 2382 was initially based mainly on the usage to be found in the *Vocabulary of Information Processing* which was established and published by the International Federation for Information Processing and the International Computation Centre, and in the *American National Dictionary for Information Processing Systems* and its earlier editions published by the American National Standards Institute (formerly known as the American Standards Association). Published and draft International Standards relating to information processing of other international organizations (such as the International Telecommunication Union and the International Electrotechnical Commission) as well as published and draft national standards have also been considered.

The purpose of ISO/IEC 2382 is to provide definitions that are rigorous, uncomplicated and which can be understood by all concerned. The scope of each concept defined has been chosen to provide a definition that is suitable for general application. In those circumstances where a restricted application is concerned, the definition may need to be more specific.

However, while it is possible to maintain the self-consistency of individual parts, the reader is warned that the dynamics of language and the problems associated with the standardization and maintenance of vocabularies may introduce duplications and inconsistencies between parts.

Introduction

Le traitement de l'information est à l'origine de multiples échanges intellectuels et matériels sur le plan international. Ceux-ci souffrent souvent des difficultés provoquées par la diversité des termes utilisés pour exprimer la même notion dans des langues ou dans des domaines différents, ou encore de l'absence ou de l'imprécision des définitions pour les notions les plus utiles.

Pour éviter des malentendus et faciliter de tels échanges, il paraît essentiel de préciser les notions, de choisir les termes à employer dans les différentes langues et dans les divers pays pour exprimer la même notion, et d'établir pour ces termes des définitions équivalentes dans chaque langue.

L'ISO 2382 a été basée à l'origine principalement sur l'usage tel qu'il a été relevé, d'une part, dans le *Vocabulary of Information Processing* établi et publié par l'International Federation for Information Processing et le Centre International de Calcul et, d'autre part, dans l'*American National Dictionary for Information Processing systems* y compris ses éditions précédentes publiées par l'American National Standards Institute (connu auparavant sous l'appellation d'American Standards Association). Les Normes internationales publiées ou au stade de projets concernant le traitement de l'information émanant d'autres organisations internationales (telles que l'Union internationale des télécommunications et la Commission électrotechnique internationale) ainsi que les normes nationales publiées ou au stade de projets, ont également été prises en compte.

Le but de l'ISO/IEC 2382 est de procurer des définitions rigoureuses, simples et compréhensibles pour tous les intéressés. La portée de chaque notion a été choisie de façon que sa définition puisse avoir la valeur la plus générale. Cependant, il est parfois nécessaire de restreindre une notion à un domaine plus étroit et de lui donner alors une définition plus spécifique.

D'autre part, si l'on peut assurer la cohérence interne de chaque partie prise individuellement, la cohérence des diverses parties entre elles est plus difficile à atteindre. Le lecteur ne doit pas s'en étonner : la dynamique des langues et les problèmes de l'établissement et de la révision des normes de vocabulaires peuvent être à l'origine de quelques répétitions ou contradictions entre des parties qui ne sont pas toutes préparées et publiées simultanément.

This page intentionally left blank

IECNORM.COM : Click to view the full PDF of ISO/IEC 2382-7:1989

Information technology — Vocabulary —

Part 07: Computer programming

Section 1: General

1.1 Scope

This part of ISO/IEC 2382 is intended to facilitate international communication in information processing. It presents, in two languages, terms and definitions of selected concepts relevant to the field of information processing and identifies relationships between the entries.

In order to facilitate their translation into other languages, the definitions are drafted so as to avoid, as far as possible, any peculiarity attached to a language.

This part of ISO/IEC 2382 defines concepts relating to programming context and some programming languages, to preparation, execution, debugging and verification of programs.

1.2 Normative references

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 2382. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this part of ISO/IEC 2382 are encouraged to investigate the possibility of applying the most recent editions of the standards indicated below. Members of IEC and ISO maintain registers of currently valid International Standards.

ISO 1087 : —¹⁾, *Terminology — Vocabulary*.

ISO 3166 : 1988, *Codes for the representation of names of countries*.

1) To be published.

Technologies de l'information — Vocabulaire —

Partie 07: Programmation des ordinateurs

Section 1: Généralités

1.1 Domaine d'application

La présente partie de l'ISO/CEI 2382 a pour objet de faciliter les échanges internationaux dans le domaine des systèmes de traitement de l'information. À cet effet, elle présente un ensemble bilingue de termes et de définitions ayant trait à des notions choisies dans ce domaine, et définit les relations pouvant exister entre les différentes notions.

Les définitions ont été établies de manière à éviter les particularismes propres à une langue donnée, en vue de faciliter leur transposition dans les langues autres que celles ayant servi à la rédaction initiale.

La présente partie de l'ISO/CEI 2382 définit les notions relatives à la programmation et à certains langages de programmation, à la préparation, à l'exécution, à la mise au point et à la vérification de programmes.

1.2 Références normatives

Les normes suivantes contiennent des dispositions qui, par suite de la référence qui en est faite, constituent des dispositions valables pour la présente partie de l'ISO/CEI 2382. Au moment de la publication, les éditions indiquées étaient en vigueur. Toute norme est sujette à révision et les parties prenantes des accords fondés sur la présente partie de l'ISO/CEI 2382 sont invitées à rechercher la possibilité d'appliquer les éditions les plus récentes des normes indiquées ci-après. Les membres de la CEI et de l'ISO possèdent le registre des Normes internationales en vigueur à un moment donné.

ISO 1087 : —¹⁾, *Terminologie — Vocabulaire*.

ISO 3166 : 1988, *Codes pour la représentation des noms de pays*.

1) À publier.

1.3 Principles and rules followed

1.3.1 Definition of an entry

Section 2 comprises a number of entries. Each entry consists of a set of essential elements that includes an index number, one term or several synonymous terms, and a phrase defining one concept. In addition, an entry may include examples, notes or illustrations to facilitate understanding of the concept.

Occasionally, the same term may be defined in different entries, or two or more concepts may be covered by one entry, as described in 1.3.5 and 1.3.8 respectively.

Other terms such as **vocabulary**, **concept**, **term** and **definition**, are used in this part of ISO/IEC 2382 with the meaning defined in ISO 1087.

1.3.2 Organization of an entry

Each entry contains the essential elements defined in 1.3.1 and, if necessary, additional elements. The entry may contain the following elements in the following order:

- a) an index number (common for all languages in which this part of ISO/IEC 2382 is published);
- b) the term or the generally preferred term in the language. The absence of a generally accepted term for the concept in the language is indicated by a symbol consisting of five points (.....); a row of dots may be used to indicate, in a term, a word to be chosen in each particular case;
- c) the preferred term in a particular country (identified according to the rules of ISO 3166);
- d) the abbreviation for the term;
- e) permitted synonymous term(s);
- f) the text of the definition (see 1.3.4);
- g) one or more examples with the heading "Example(s)";
- h) one or more notes specifying particular cases in the field of application of the concepts, with the heading "NOTE(S)";
- i) a picture, a diagram, or a table which could be common to several entries.

1.3.3 Classification of entries

A two-digit serial number is assigned to each part of this International Standard, beginning with 01 for "fundamental terms".

The entries are classified in groups to each of which is assigned a four-digit serial number, the first two digits being those of the part of this International Standard.

1.3 Principes d'établissement et règles suivies

1.3.1 Définition de l'article

La section 2 est composée d'un certain nombre d'articles. Chaque article est composé d'un ensemble d'éléments essentiels comprenant le numéro de référence, le terme ou plusieurs termes synonymes et la définition d'une notion couverte par ces termes. Cet ensemble peut être complété par des exemples, des notes, des schémas ou des tableaux destinés à faciliter la compréhension de la notion.

Parfois, le même terme peut être défini dans des articles différents, ou bien deux notions ou davantage peuvent être couvertes par un seul article : voir respectivement en 1.3.5 et 1.3.8.

D'autres termes tels que **vocabulaire**, **notion**, **terme**, **définition** sont employés dans la présente partie de l'ISO/CEI 2382 avec le sens qui leur est donné dans l'ISO 1087.

1.3.2 Constitution d'un article

Chaque article contient les éléments essentiels définis en 1.3.1 et, si nécessaire, des éléments supplémentaires. L'article peut donc comprendre dans l'ordre les éléments suivants:

- a) un numéro de référence (le même, quelle que soit la langue de publication de la présente partie de l'ISO/CEI 2382);
- b) le terme, ou le terme préféré en général dans la langue. L'absence, dans une langue, de terme consacré ou à conseiller pour exprimer une notion est indiquée par un symbole consistant en cinq points de suspension (.....); les points de suspension peuvent être employés pour désigner, dans un terme, un mot à choisir dans chaque cas particulier;
- c) le terme préféré dans un certain pays (identifié selon les règles de l'ISO 3166);
- d) l'abréviation pouvant être employée à la place du terme;
- e) le terme ou les termes admis comme synonymes;
- f) le texte de la définition (voir 1.3.4);
- g) un ou plusieurs exemples précédés du titre «Exemple(s)»;
- h) une ou plusieurs notes précisant le domaine d'application de la notion, précédées du titre «NOTE(S)»;
- i) une figure, un schéma ou un tableau, pouvant être communs à plusieurs articles.

1.3.3 Classification des articles

Chaque partie de la présente Norme internationale reçoit un numéro d'ordre à deux chiffres, en commençant pas 01 pour le chapitre «termes fondamentaux».

Les articles sont répartis en groupes qui reçoivent chacun un numéro d'ordre à quatre chiffres, les deux premiers chiffres étant ceux du numéro de partie de la présente Norme internationale.

Each entry is assigned a six-digit index number, the first four digits being those of the part of this International Standard and the group.

In order that versions of this International Standard in various languages are related, the numbers assigned to parts, groups and entries are the same for all languages.

1.3.4 Selection of terms and wording of definitions

The selection of terms and the wording of definitions have, as far as possible, followed established usage. When there were contradictions, solutions agreeable to the majority have been sought.

1.3.5 Multiple meanings

When, in one of the working languages, a given term has several meanings, each meaning is given a separate entry in order to facilitate translation into other languages.

1.3.6 Abbreviations

As indicated in 1.3.2, abbreviations in current use are given for some terms. Such abbreviations are not used in the texts of the definitions, examples or notes.

1.3.7 Use of parentheses

In some terms, a word or words printed in bold typeface are placed between parentheses. These words are part of the complete term, but they may be omitted when use of the abridged term in a technical context does not introduce ambiguity. In the text of another definition, example, or note of ISO/IEC 2382, such a term is used only in its complete form.

In some entries, the terms are followed by words in parentheses in normal typeface. These words are not a part of the term but indicate directives for the use of the term, its particular field of application, or its grammatical form.

1.3.8 Use of brackets

When several closely related terms can be defined by texts that differ only in a few words, the terms and their definitions are grouped in a single entry. The words to be substituted in order to obtain the different meanings are placed in brackets, i.e. [], in the same order in the term and in the definition. In order to avoid uncertainty regarding the words to be substituted, the last word that according to the above rule could be placed in front of the opening bracket is, wherever possible, placed inside the bracket and repeated for each alternative.

Chaque article est repéré par un numéro de référence à six chiffres, les quatre premiers chiffres étant ceux du numéro de partie de la présente Norme internationale et de groupe.

Les numéros des parties, des groupes et des articles sont les mêmes pour toutes les langues, afin de mettre en évidence les correspondances des versions de la présente Norme internationale.

1.3.4 Choix des termes et des définitions

Les choix qui ont été faits pour les termes et leurs définitions sont, dans toute la mesure du possible, compatibles avec les usages établis. Lorsque certains usages apparaissent contradictoires, des solutions de compromis ont été retenues.

1.3.5 Pluralité de sens ou polysémie

Lorsque, dans l'une des langues de travail, un même terme peut prendre plusieurs sens, ces sens sont définis dans des articles différents, pour faciliter l'adaptation du vocabulaire dans d'autres langues.

1.3.6 Abréviations

Comme indiqué en 1.3.2, des abréviations littérales d'usage courant, au moins en anglais, sont indiquées pour certains termes. De telles abréviations ne sont pas employées dans le corps des définitions, exemples ou notes.

1.3.7 Emploi des parenthèses

Dans certains termes, un ou plusieurs mots imprimés en caractères gras sont placés entre parenthèses. Ces mots font partie intégrante du terme complet, mais peuvent être omis lorsque le terme ainsi abrégé peut être employé dans un contexte technique déterminé sans que cette omission introduise d'ambiguïté. Un tel terme n'est employé dans le texte d'une autre définition, d'un exemple ou d'une note, de l'ISO/CEI 2382, que sous sa forme complète.

Dans certains articles, les termes définis sont suivis par des expressions imprimées en caractères normaux et placées entre parenthèses. Ces expressions ne font pas partie du terme mais indiquent des prescriptions d'emploi, précisent un domaine d'application particulier ou indiquent une forme grammaticale.

1.3.8 Emploi des crochets

Lorsque plusieurs termes étroitement apparentés peuvent être définis par des textes presque identiques, à quelques mots près, les termes et leurs définitions ont été groupés en un seul article. Les mots à substituer à ceux qui les précèdent pour obtenir les différents sens sont placés entre crochets (c'est-à-dire []) dans le même ordre dans le terme et dans la définition. En vue d'éviter toute incertitude sur les mots à remplacer, le dernier mot qui, suivant la règle ci-dessus pourrait être placé devant le crochet d'ouverture, est placé, si possible, à l'intérieur des crochets et répété à chaque occasion.

1.3.9 Use of terms printed in italic typeface in definitions and the use of an asterisk

A term printed in italic typeface in a definition, an example, or a note is defined in another entry in this International Standard, which may be in another part. However, the term is printed in italic typeface only the first time it occurs in each entry.

Italic typeface is also used for other grammatical forms of a term, for example, plurals of nouns and participles of verbs.

The basic forms of terms printed in italic typeface which are defined in this part of ISO/IEC 2382 are listed in the index at the end of the part (see 1.3.11).

An asterisk is used to separate terms printed in italic typeface when two such terms are referred to in separate entries and directly follow each other (or are separated only by a punctuation sign).

Words or terms that are printed in normal typeface are to be understood as defined in current dictionaries or authoritative technical vocabularies.

1.3.10 Spelling

In the English language version of this part of ISO/IEC 2382, terms, definitions, examples, and notes are given in the spelling preferred in the USA. Other correct spellings may be used without violating this part of ISO/IEC 2382.

1.3.11 Organization of the alphabetical index

For each language used, an alphabetical index is provided at the end of each part. The index includes all terms defined in the part. Multiple-word terms appear in alphabetical order under each of their key words.

1.3.9 Emploi dans les définitions de termes imprimés en caractères italiques et de l'astérisque

Dans le texte d'une définition, d'un exemple ou d'une note, tout terme imprimé en caractères italiques a le sens défini dans un autre article de la présente Norme internationale, qui peut se trouver dans une autre partie. Cependant, le terme est imprimé en caractères italiques uniquement la première fois qu'il apparaît dans chaque article.

Les caractères italiques sont également utilisés pour les autres formes grammaticales du terme, par exemple, les noms au pluriel et les verbes au participe.

La liste des formes de base des termes imprimés en caractères italiques qui sont définis dans la présente partie de l'ISO/CEI 2382 est fournie dans l'index à la fin de la partie (voir 1.3.11).

L'astérisque sert à séparer les termes imprimés en caractères italiques quand deux termes se rapportent à des articles séparés et se suivent directement (ou bien sont séparés simplement par un signe de ponctuation).

Les mots ou termes imprimés en caractères normaux doivent être compris dans le sens qui leur est donné dans les dictionnaires courants ou vocabulaires techniques faisant autorité.

1.3.10 Mode d'écriture et orthographe

Dans la version anglaise de la présente partie de l'ISO/CEI 2382, les termes, définitions, exemples et notes sont écrits suivant l'orthographe prévalant aux États-Unis. D'autres orthographes correctes peuvent être utilisées sans violer la présente partie de l'ISO/CEI 2382.

1.3.11 Constitution de l'index alphabétique

Pour chaque langue de travail, un index alphabétique est fourni à la fin de chaque partie. L'index comprend tous les termes définis dans la partie. Les termes composés de plusieurs mots sont répertoriés alphabétiquement suivant chacun des mots constituants caractéristiques ou mots clés.

Section 2: Terms and definitions

07 Computer programming

07.01 General terms

07.01.01

programming environment

An integrated collection of *software* and *hardware* to support the development of *computer programs*.

07.01.02

programming system

In a *programming environment*, the *software* required for the development and use of *computer programs*.

07.01.03

(program) library

An organized collection of *computer programs*, or parts of computer programs, and possibly information pertaining to their use.

NOTE — A program library is often called according to the characteristic of its elements, for example, a *procedure* library, a *source program* library.

07.01.04

utility routine [program]

service routine [program]

A *routine* [A *computer program*] that provides general, frequently needed services for *computer* users and service personnel.

Examples: An *input routine*; a *diagnostic program*; a *trace program*; a *sort program*.

07.01.05

algorithmic language

An artificial language for expressing *algorithms*.

07.01.06

programming language

An artificial language for expressing *computer programs*.

07.01.07

problem-oriented language

A *programming language* that reflects the concepts of a particular application area.

Examples: SQL for *database* applications, COBOL for business applications.

07.01.08

machine language

An artificial language composed of the *machine instructions* of a *computer*.

Section 2: Termes et définitions

07 Programmation des ordinateurs

07.01 Termes généraux

07.01.01

contexte de programmation

environnement de programmation

Ensemble intégré de *matériel* et de *logiciel* permettant le développement de *programmes d'ordinateur*.

07.01.02

système de programmation

Dans un *contexte de programmation*, **logiciel* nécessaire au développement et à l'utilisation de *programmes d'ordinateur*.

07.01.03

bibliothèque (de programmes)

programmathèque

Ensemble organisé de *programmes d'ordinateur* ou de parties de programmes d'ordinateur, associés éventuellement à toute information relative à leur utilisation.

NOTE — Une bibliothèque de programmes est souvent qualifiée par la caractéristique de ses éléments, par exemple une bibliothèque de *procédures*, une bibliothèque de *programmes d'origine*.

07.01.04

routine [programme] de service

routine [programme] utilitaire

Routine [*Programme d'ordinateur*] mettant à la disposition des utilisateurs d'*ordinateurs* et du personnel informatique des fonctions générales utilisées fréquemment.

Exemples: *Programme d'entrée*, **programme de diagnostic*, **programme de traçage*, **programme de tri*.

07.01.05

langage algorithmique

Langage artificiel permettant d'exprimer des *algorithmes*.

07.01.06

langage de programmation

Langage artificiel permettant d'exprimer des *programmes d'ordinateur*.

07.01.07

langage adapté aux problèmes

langage orienté (vers les) problèmes

Langage de programmation qui convient pour représenter un domaine d'application particulier.

Exemples: SQL pour les applications de *bases de données*, COBOL pour les applications de gestion.

07.01.08

langage-machine

Langage artificiel dont les éléments sont des *instructions-machine*.

07.01.09

assembly language

A *programming language* that provides symbolic forms for *machine instructions*.

NOTE — Assembly languages usually also provide *macro calls*.

07.01.10

high-level language

A *programming language* whose concepts and structures are convenient for human reasoning.

NOTE — High-level languages are independent of the structures of *computers* and *operating systems*.

Example: Pascal.

07.01.11

declarative language

A *programming language* for expressing *declarations*.

Example: A *data description language*.

07.01.12

procedural language

A *programming language* in which computations are expressed in terms of *statement *sequences*.

Example: Pascal.

07.01.13

functional language

A *programming language* in which computations are expressed in terms of *function procedure *calls*.

Example: LISP.

07.01.14

object-oriented language

A *programming language* that reflects the concepts of *object-oriented programming*.

Example: SMALLTALK.

07.01.15

microprogram

A *sequence* of *microinstructions*.

NOTE — Microprograms are mainly used to implement *machine instructions*.

07.02 Programming concepts

07.02.01

structured programming

A method for constructing *programs* using only hierarchically *nested* constructs each having a single entry and a single exit point.

NOTE — Three types of *control flow* are used in structured programming: sequential, conditional, and iterative.

07.01.09

langage d'assemblage

Langage de programmation permettant de disposer d'*instructions-machine* sous forme symbolique.

NOTE — Un langage d'assemblage peut également comprendre des *macro-instructions*.

07.01.10

langage évolué

Langage de programmation dont les concepts et les structures sont relativement proches du raisonnement humain.

NOTE — Les langages évolués sont indépendants de la structure des *calculateurs* et des *systèmes d'exploitation*.

Exemple: Pascal.

07.01.11

langage déclaratif

Langage de programmation conçu pour exprimer des *déclarations*.

Exemple: Un *langage de description de données*.

07.01.12

langage de procédure

langage procédural

Langage de programmation dans lequel les calculs sont exprimés sous forme de *séquences d'instructions*.

Exemple: Pascal.

07.01.13

langage fonctionnel

Langage de programmation dans lequel les calculs sont exprimés sous forme d'*appels de fonctions*.

Exemple: LISP.

07.01.14

langage adapté à l'objet

langage orienté (vers l')objet

Langage de programmation basé sur les concepts de la *programmation adaptée à l'objet*.

Exemple: SMALLTALK.

07.01.15

microprogramme

Séquence de micro-instructions

NOTE — Un microprogramme s'utilise essentiellement pour mettre en œuvre des *instructions-machine*.

07.02 Notions liées à la programmation

07.02.01

programmation structurée

Méthode permettant de construire des *programmes* en n'utilisant que des constructions hiérarchiquement *emboîtées* ayant chacune un seul point d'entrée et un seul point de sortie.

NOTE — La programmation structurée emploie trois types de *flux de commande*: séquentiel, conditionnel et itératif.

07.02.02**stepwise refinement**

A method for constructing *programs* in successive steps such that at each step an action is expressed in terms of more primitive actions.

07.02.03**to nest**

To incorporate one or more structures of one kind into a structure of the same kind.

Exemples: To nest one *loop* (the nested or inner loop) within another loop (the nesting or outer loop); to nest one *subroutine* within another subroutine.

07.02.04**functional programming**

A method for structuring *programs* mainly as *sequences* of possibly *nested *function procedure *calls*.

07.02.05**object-oriented programming**

A method for structuring *programs* as hierarchically organized classes describing the *data* and *operations* of objects that may interact with other objects.

07.02.06**logic programming**

A method for structuring *programs* as sets of logical rules with predefined algorithms for the processing of *input data* to a program according to the rules of that program.

07.02.07**1. microprogramming**

The preparation of *microprograms*.

07.02.08**2. microprogramming**

The technique used in the design of *hardware* that is to be controlled by *microprograms*.

07.02.09**subroutine**

A *sequence* of *instructions* whose *execution* is invoked by a *call*.

07.02.10**recursive subroutine**

A *subroutine* that may invoke itself.

NOTE — A recursive subroutine normally contains a *call* that invokes this subroutine directly or indirectly.

07.02.11**coroutine**

A *subroutine* which, when called again after an *execution*, resumes at the *return* point of its previous execution.

07.02.02**affinement progressif**

Méthode permettant de construire des *programmes* par étapes successives, de telle sorte qu'à chaque étape on exprime une action sous la forme d'actions plus élémentaires.

07.02.03**emboîter**

Incorporer plusieurs structures d'un certain type à l'intérieur d'une structure du même type.

Exemples: Placer une *boucle* (boucle emboîtée ou interne) à l'intérieur d'une autre boucle (boucle emboîtante ou externe); placer un *sous-programme* à l'intérieur d'un autre sous-programme.

07.02.04**programmation fonctionnelle**

Méthode utilisée pour structurer les *programmes* en les écrivant, pour la plus grande part, sous la forme de suites d'*appels* de *fonctions*, éventuellement *emboîtés*.

07.02.05**programmation adaptée à l'objet****programmation orientée (vers l')objet**

Méthode utilisée pour structurer les *programmes* sous la forme de classes organisées hiérarchiquement et décrivant les *données* et les *opérations* portant sur des objets qui peuvent interagir avec d'autres objets.

07.02.06**programmation en logique**

Méthode utilisée pour structurer les *programmes* sous la forme d'ensembles de règles logiques munis d'algorithmes prédéfinis permettant de traiter, selon ces règles, les *données d'entrée* d'un programme.

07.02.07**1. microprogrammation**

Préparation de *microprogrammes*.

07.02.08**2. microprogrammation**

Technique utilisée dans la conception du *matériel* commandé par *microprogramme*.

07.02.09**sous-programme**

Séquence d'instructions dont l'*exécution* est lancée par un *appel*.

07.02.10**sous-programme récursif**

Sous-programme qui peut s'appeler lui-même.

NOTE — Un sous-programme récursif contient normalement un *appel* qui lance l'*exécution* de ce sous-programme, directement ou indirectement.

07.02.11**coroutine**

Sous-programme qui, lorsqu'il est rappelé après une *exécution*, reprend au point de *retour* de sa précédente exécution.

07.02.12

loop

A *sequence of instructions* that is to be *executed* iteratively.

07.02.13

infinite loop

closed loop /US/

A *loop* whose *execution* can be terminated only by intervention from outside the *computer program* in which the loop is included.

07.02.14

jump

A departure from the sequential *execution of instructions*.

07.02.15

flag

A *variable* indicating that a certain condition holds.

07.02.16

switch

A choice of one *jump* from a selection of jumps, controlled by a *flag*.

07.02.17

work(ing) space

work(ing) area

A portion of *storage* used by a *computer program* to hold *data* temporarily.

07.02.18

hashing

A method of *transforming* a *search key* into an *address* for the purpose of *storing* and retrieving items of *data*.

NOTE — The method is often designed to minimize the *search time*.

07.02.19

synchronization

The action of forcing certain points in the *execution sequences* of two or more *asynchronous procedures* to coincide in time.

07.02.20

mutual exclusion

A principle requiring that, at a given time, only one *asynchronous procedure* may access the same *shared variable* or *execute* members of a group of *critical sections*.

07.02.21

semaphore

A *variable* that is used to enforce *mutual exclusion*.

07.02.12

boucle

Suite d'*instructions* dont l'*exécution* doit être répétée.

07.02.13

boucle fermée

Boucle dont on ne peut sortir que par une intervention extérieure au *programme d'ordinateur* contenant cette boucle.

07.02.14

saut

Déviation par rapport à l'*exécution séquentielle* des *instructions*.

07.02.15

drapeau

indicateur d'aiguillage

Variable indiquant qu'une condition déterminée est vérifiée ou non.

07.02.16

aiguillage

Choix commandé par un *drapeau*, d'un *saut* parmi plusieurs sauts possibles.

07.02.17

espace de travail

zone de manœuvre

zone de travail

Partie de *mémoire* utilisée par un *programme d'ordinateur* pour y ranger temporairement des *données*.

07.02.18

adressage calculé

tronçonnage

Méthode utilisée pour *transformer* une *clé de recherche* en une *adresse*, en vue de ranger ou de retrouver des *données*.

NOTE — Cette méthode vise à minimiser le *temps d'exploration*.

07.02.19

synchronisation

Action de forcer certains points dans les *séquences d'exécution* de plusieurs *procédures asynchrones*, pour qu'ils coïncident dans le temps.

07.02.20

exclusion mutuelle

Principe selon lequel une seule *procédure asynchrone* à la fois accède à une *variable partagée* ou *exécute* des membres d'un groupe de *sections critiques*.

07.02.21

sémaphore

Variable employée pour imposer une *exclusion mutuelle*.

07.02.22**producer**

An *asynchronous procedure* that provides *data* to be used by other asynchronous procedures.

07.02.23**consumer**

An *asynchronous procedure* that uses *data* provided by other asynchronous procedures.

07.02.24**shared variable**

A *variable* that can be accessed by two or more *asynchronous procedures* or concurrently executed **computer programs*.

07.02.25**to bind**

To relate an *identifier* to another object in a *program*.

Exemples: To relate an identifier to a value, an *address* or another identifier, or to associate *formal parameters* and *actual parameters*.

07.02.26**macro call**

A *statement*, embedded in a *source language*, that is to be replaced by a defined *statement sequence* in the same *source language*.

NOTE — The macro call will also specify the *actual parameters* for any *formal parameters* in the *macrodefinition*.

07.02.27**macrodefinition**

A possibly parameterized specification for a *statement *sequence* to replace a *macro call*.

NOTE — A macrodefinition may be considered as a *procedure* to be executed by a *macrogenerator* yielding the *statement sequence*.

07.03 Program preparation**07.03.01****to translate** (in programming languages)

To *transform* all or part of a *program* expressed in one *programming language*, into another programming language or into a *machine language* suitable for *execution*.

07.03.02**to assemble**

To *translate* an *assembly language *program* into an *object program*.

07.03.03**to compile**

To *translate* all or part of a *program* expressed in a *high-level language* into a *computer program* expressed in an *intermediate language*, an *assembly language*, or a *machine language*.

07.02.22**producteur**

Procédure asynchrone qui fournit des *données* pour utilisation par d'autres procédures asynchrones.

07.02.23**consommateur**

Procédure asynchrone qui utilise des *données* fournies par d'autres procédures asynchrones.

07.02.24**variable partagée**

Variable à laquelle peuvent accéder plusieurs *procédures asynchrones*, ou bien plusieurs *programmes d'ordinateur *exécutés* concurremment.

07.02.25**relier**

Rattacher un *identificateur* à un autre objet de *programme*.

Exemples: Relier un identificateur à une valeur, à une *adresse* ou à un autre identificateur ou bien relier des *paramètres formels* à des *paramètres réels*.

07.02.26**macro-instruction**

Dans un *langage d'origine*, **instruction* destinée à être remplacée par une *séquence d'instructions* dans le même langage d'origine.

NOTE — La macro-instruction spécifie également un *paramètre réel* pour chacun des *paramètres formels* de la *macrodéfinition*.

07.02.27**macrodéfinition**

Spécification, éventuellement paramétrée, d'une *séquence d'instructions* destinée à remplacer une *macro-instruction*.

NOTE — Une macrodéfinition peut être considérée comme une *procédure* qu'un *macrogénérateur* doit *exécuter* pour produire la séquence d'instructions.

07.03 Préparation des programmes**07.03.01****traduire** (en langages de programmation)

Transformer tout ou partie d'un *programme* écrit dans un *langage de programmation*, pour l'exprimer dans un autre langage de programmation ou dans un *langage-machine* convenant à l'*exécution*.

07.03.02**assembler**

Traduire un *programme* écrit en *langage d'assemblage* en un *programme-objet*.

07.03.03**compiler**

Traduire tout ou partie d'un *programme* écrit en *langage évolué* en un *programme d'ordinateur* exprimé en *langage intermédiaire*, en *langage d'assemblage* ou en *langage-machine*.

07.03.04

to interpret

To analyse and execute each *statement* in a *source program* before *translating* and executing the next *statement*.

07.03.05

translator

translation program

A *computer program* that can *translate*.

07.03.06

assembler

A *translator* that can *assemble*.

07.03.07

compiler

A *translator* that can *compile*.

07.03.08

interpreter

interpretive program

A *computer program* that can *interpret*.

07.03.09

source language

The *programming language* for expressing *source programs* that a particular *translator* can accept.

07.03.10

target language

The *output language* of a *translator*.

07.03.11

object language

A *target language* for expressing *objects programs*.

07.03.12

intermediate language

A language into which a *source program* or *statement* is *translated* before it is further translated or *interpreted*.

07.03.13

source program

A *program* that a particular *translator* can accept.

07.03.14

target program

The *translated* version of a *source program*.

07.03.15

object program

A *target program* suitable for *execution*.

NOTE — An object program may or may not require *linking*.

07.03.04

interpréter

Analyser et exécuter chaque *instruction* d'un *programme d'origine* avant de *traduire* et d'exécuter l'*instruction* suivante.

07.03.05

traducteur

programme de traduction

Programme d'ordinateur conçu pour *traduire*.

07.03.06

assembleur

Traducteur conçu pour *assembler*.

07.03.07

compilateur

Traducteur conçu pour *compiler*.

07.03.08

interpréteur

programme interprétatif

interprète

Programme d'ordinateur conçu pour *interpréter*.

07.03.09

langage d'origine

langage-source

Langage de programmation conçu pour exprimer des *programmes d'origine* que l'on peut soumettre à un *traducteur* déterminé.

07.03.10

langage résultant

Langage obtenu en *sortie* d'un *traducteur*.

07.03.11

langage-objet

Langage résultant conçu pour exprimer des *programmes-objets*.

07.03.12

langage intermédiaire

Langage dans lequel est *traduit* un *programme d'origine* ou une *instruction* avant d'être *retraduit* ou *interprété*.

07.03.13

programme d'origine

programme-source

Programme que l'on peut soumettre à un *traducteur* déterminé.

07.03.14

programme résultant

Version résultant de la traduction d'un *programme d'origine*.

07.03.15

programme-objet

Programme résultant convenant à l'*exécution*.

NOTE — Ce programme peut ou non exiger une édition de liens.

07.03.16**1. translation [compilation] [assembly] time**

Any instant at which *translation* [compilation] [assembly] takes place.

07.03.17**2. translation [compilation] [assembly] time**

The amount of time needed to *translate* [compile] [assemble] a *program*.

07.03.18**translator [assembler] [compiler] [interpreter] directive**

A language construct for controlling the *translation* [assembly] [compilation] [interpretation] of a *program*.

07.03.19**source module****compilation unit**

All or part of a *source program* sufficiently complete for compilation.

07.03.20**dependent compilation****separate compilation** (deprecated)

The compilation of a *source module* using all the necessary interface and context information from related source modules.

NOTE — Interface and context information is used by the *compiler* to check validity and to resolve *references*.

07.03.21**independent compilation****separate compilation** (deprecated)

The compilation of a *source module* not using the interface and context information from related source modules.

NOTE — When independently compiled units are eventually combined, it may be necessary to check interface and context information for validity.

07.03.22**generic unit**

A possibly parameterized model of a language construct from which, at *translation time*, a language construct proper is derived.

NOTE — There is some analogy between generic units and *macro-definitions*: the language construct derived from the generic unit corresponds to the *statement* *sequence replacing a *macro call* and is referred to as a "generic instantiation".

07.03.23**macrogenerator****macroprocessor**

A *program* for replacing *macro calls* with defined *statement* *sequences according to *macrodefinitions*.

NOTE — A macrogenerator may be an independent *computer program* or it may be integrated as a subprogram in a *compiler* or *assembler* to generate the *source program*.

07.03.16**1. temps de traduction [de compilation] [d'assemblage]**

Tout instant où se déroule une *traduction* [une compilation] [un assemblage].

07.03.17**2. temps de traduction [de compilation] [d'assemblage]**

Durée nécessaire pour *traduire* [compiler] [assembler] un *programme*.

07.03.18**directive de traduction [d'assemblage] [de compilation] [d'interprétation]**

Élément de langage fournissant des indications pour *traduire* [assembler] [compiler] [interpréter] un *programme*.

07.03.19**module-source****unité de compilation**

Tout ou partie d'un *programme d'origine*, qui constitue un tout se prêtant à une compilation correcte.

07.03.20**compilation dépendante****compilation séparée**

Compilation d'un *module-source* utilisant toutes les informations nécessaires d'interface et de contexte fournies par les modules-source associés.

NOTE — Les informations d'interface et de contexte sont utilisées par le *compilateur* pour vérifier la validité et pour résoudre les *références*.

07.03.21**compilation indépendante**

Compilation d'un *module-source* n'utilisant pas les informations d'interface et de contexte fournies par les modules-source associés.

NOTE — Quand des unités compilées de façon indépendante sont ultérieurement combinées, il convient d'en vérifier la validité au moyen des informations d'interface et de contexte.

07.03.22**unité générique**

Modèle éventuellement paramétré d'un élément de langage, dont est dérivé, au moment de la traduction, un élément de langage conforme.

NOTE — Il y a une certaine analogie entre unités génériques et *macro-définitions*: l'élément de langage dérivé de l'unité générique correspond à la *séquence d'instructions* remplaçant une *macro-instruction*.

07.03.23**macrogénérateur****macroprocesseur**

Programme conçu pour remplacer des *macro-instructions* par des *séquences* définies d'*instructions*, selon des *macrodéfinitions*.

NOTE — Un macrogénérateur peut être un *programme d'ordinateur* indépendant ou être intégré en tant que *sous-programme* dans un *compilateur* ou dans un *assembleur* pour générer le *programme d'origine*.

07.03.24

(program) generator

A *computer program* that can produce other computer programs.

07.03.25

(language) preprocessor

A *functional unit* that effects preparatory processing of *computer programs*.

Exemple: A *macrogenerator* may function as a preprocessor of a *translator*.

07.03.26

object module

All or part of an *object program* sufficiently complete for *linking*.

NOTE — *Assemblers* and *compilers* usually produce object modules.

07.03.27

load module

All or part of a *computer program* in a form suitable for loading into *main storage* for *execution*.

NOTE — A load module is usually the output of a *linkage editor*.

07.03.28

segment

A portion of a *computer program* that may be *executed* without the entire computer program being *resident* in *main storage*.

07.03.29

overlay (segment)

Each of several *segments* of a *computer program* that, one at a time, occupy the same area of *main storage*, when *executed*.

07.03.30

resident

Pertaining to *computer programs* or *data* while they remain on a particular *storage device*.

07.03.31

to link

To interconnect items of *data* or portions of one or more *computer programs*.

Exemples: The linking of *object programs* by a *linkage editor*. The linking of data items by *pointers*.

07.03.32

linkage editor

linker

A *computer program* for creating *load modules* from one or more *object modules* or *load modules* by resolving cross references among the modules and, if necessary, by adjusting *addresses*.

07.03.24

générateur (de programmes)

Programme d'ordinateur conçu pour produire d'autres programmes d'ordinateur.

07.03.25

préprocesseur

Unité fonctionnelle qui effectue des tâches préalables au traitement de *programmes d'ordinateur*.

Exemple: Un *macrogénérateur* peut tenir lieu de préprocesseur pour un *traducteur*.

07.03.26

module-objet

Tout ou partie d'un *programme-objet* suffisamment complet pour être soumis à l'*éditeur de liens*.

NOTE — *Assembleurs* et *compilateurs* produisent habituellement des modules-objets.

07.03.27

module chargeable

Tout ou partie d'un *programme d'ordinateur*, écrit sous une forme qui permet de le charger dans la *mémoire principale*, pour *exécution*.

NOTE — Un module chargeable est généralement produit par un *éditeur de liens*.

07.03.28

segment

Partie d'un *programme d'ordinateur* pouvant être *exécutée* sans que la totalité de ce programme se trouve en *mémoire principale*.

07.03.29

segment en recouvrement

Chacun des *segments* d'un *programme d'ordinateur* qui occupent tour à tour le même emplacement en *mémoire principale* durant leur *exécution*.

07.03.30

résidant (adjectif)

Qualifie des *programmes d'ordinateur* ou des *données*, dans la mesure où ils se trouvent dans une *mémoire* déterminée.

07.03.31

lier

Interconnecter des *données* ou des parties d'un ou de plusieurs *programmes d'ordinateur*.

Exemples: La liaison de *programmes-objets* par un *éditeur de liens*. La liaison de données par des *pointeurs*.

07.03.32

éditeur de liens

Programme d'ordinateur conçu pour créer un *module chargeable* à partir d'un ou de plusieurs *modules-objets* ou d'autres modules chargeables, en résolvant les références croisées entre les modules et, le cas échéant, en adaptant les *adresses*.

07.04 Program execution**07.04.01****language processor**

A functional unit for translating and executing *computer programs written in a specified programming language.

Example: A LISP machine.

07.04.02**1. execution time****1. run time**

Any instant at which the execution of a particular computer program takes place.

07.04.03**2. execution time****2. run time**

The amount of time needed for the execution of a particular computer program.

07.04.04**execution profile**

A representation of the absolute or relative execution frequencies or execution times of the instructions of a computer program.

07.04.05**to relocate**

To move all or part of an object program in an address space and to make the necessary adjustment of addresses so that the object program can be executed in the new location.

07.04.06**relocatable program**

An object program that is in such a form that it may be relocated.

07.04.07**reentrant program [routine] [subroutine] [procedure]**

A computer program [A routine] [A subroutine] [A procedure] that may be entered at any time before any prior execution of the program [routine] [subroutine] [procedure] has been completed.

07.04.08**to exit**

To execute an instruction within a portion of a computer program in order to terminate the execution of that portion.

NOTE — Such portions of computer programs include loops, *subroutines, *modules, etc.

07.04.09**breakpoint**

A point in a computer program where execution may be halted.

NOTE — A breakpoint is usually at the beginning of an instruction where halts, caused by external interventions, are convenient for resuming execution.

07.04 Exécution de programmes**07.04.01****processeur de langage**

Unité fonctionnelle destinée à traduire et à exécuter des programmes d'ordinateur écrits dans un langage de programmation déterminé.

Exemple: Une machine LISP.

07.04.02**moment de l'exécution**

Tout instant où se déroule l'exécution d'un programme d'ordinateur déterminé.

07.04.03**durée d'exécution**

Temps nécessaire à l'exécution d'un programme d'ordinateur déterminé.

07.04.04**profil d'exécution**

Représentation des fréquences absolues ou relatives d'exécution, ou des débuts de l'exécution, des instructions d'un programme d'ordinateur.

07.04.05**translater**

Transférer tout ou partie d'un programme-objet dans un espace d'adresses et faire les modifications d'adresses nécessaires afin de permettre l'exécution du programme dans le nouvel emplacement.

07.04.06**programme translatable**

Programme-objet dont la forme lui permet d'être traduit.

07.04.07**programme [routine] [sous-programme] [procédure] ré-entrant**

Programme d'ordinateur [routine] [sous-programme] [procédure] dont on peut à tout moment déclencher une nouvelle exécution avant l'achèvement d'une exécution antérieure.

07.04.08**sortir**

Exécuter une instruction dans une partie de programme d'ordinateur de façon à mettre fin à l'exécution de cette partie.

NOTE — De telles parties de programmes d'ordinateur peuvent être des boucles, des sous-programmes, des modules, etc.

07.04.09**point d'interruption**

Point d'un programme d'ordinateur où l'exécution peut être interrompue.

NOTE — On place généralement le point d'interruption au début d'une instruction de telle sorte que, après un arrêt dû à une intervention extérieure, il soit facile de reprendre l'exécution à partir de cette instruction.

**07.04.10
checkpoint**

A sequence of *instructions* in a *computer program* for recording the status of *execution* for *restarting*.

NOTE — The term "checkpoint" is also used for the recorded status of execution and for "executing a checkpoint".

**07.04.11
to restart**

To resume the *execution* of a *computer program* using the *data* recorded at a *checkpoint*.

**07.04.12
to recover**

After an *execution* failure, to establish a previous or new status from which execution can be resumed.

**07.04.13
operation code trap**

A specific value that replaces the normal *operation part* of a *machine instruction* at a particular location to cause an *interrupt* when that machine instruction is *executed*.

**07.04.14
starvation**

A situation in which an *activation* of an *asynchronous procedure* is incapable of proceeding within any predictable period of time because other *concurrent* activations permanently retain required resources.

**07.04.15
deadlock**

A situation in which two or more *activations* of *asynchronous procedures* are incapable of proceeding because of their mutual dependencies.

**07.04.16
bootstrap**

A sequence of *instructions* whose *execution* causes additional instructions to be *loaded* and *executed* until a complete *computer program* is loaded into *storage*.

**07.04.17
to bootstrap**

To *execute* a *bootstrap*.

NOTE — The term bootstrapping is also used for *translating* a *compiler* by using itself or a previous version as the *translator*.

07.05 Program debugging and verification

**07.05.01
to dump**

To record, at a particular instant, the contents of all or part of one *storage device* in another *storage device*.

NOTE — Dumping is usually for the purpose of *debugging*.

**07.04.10
point de reprise
point de contrôle**

Dans un *programme d'ordinateur*, séquence d'*instructions* où l'on enregistre les *données d'exécution* en vue d'une reprise ultérieure.

**07.04.11
reprendre
redémarrer**

Poursuivre l'*exécution* d'un *programme d'ordinateur* en utilisant les *données* enregistrées à un *point de reprise*.

**07.04.12
récupérer**

Après un incident d'*exécution*, créer ou recréer des conditions à partir desquelles l'*exécution* peut *reprendre*.

**07.04.13
déroutement**

Valeur déterminée qui remplace la *partie opération* normale d'une *instruction machine* en un point particulier, afin de provoquer une *interruption* lors de l'*exécution* de cette instruction-machine.

**07.04.14
étrangement
blocage**

Situation dans laquelle l'*activation* d'une *procédure asynchrone* est bloquée en raison d'activations concurrentes qui mobilisent, pour une durée imprévisible, les ressources nécessaires.

**07.04.15
étreinte fatale
verrou mortel**

Situation dans laquelle l'interdépendance de plusieurs *activations* de *procédures asynchrones* interdit leur *exécution*.

**07.04.16
amorcer**

Suite d'*instructions* dont l'*exécution* entraîne le chargement et l'*exécution* d'autres instructions jusqu'à ce qu'un *programme d'ordinateur* complet soit chargé en *mémoire*.

**07.04.17
amorcer
Exécuter une amorce.**

NOTE — Le terme amorçage est également utilisé dans le sens de: *traduire* un *compilateur* en l'utilisant, tel quel ou dans l'une de ses versions antérieures, comme *traducteur*.

07.05 Mise au point et vérification des programmes

**07.05.01
vider
clicher**

Écrire, à un instant déterminé, le contenu de tout ou partie d'une *mémoire* sur un autre *support de données*.

NOTE — Le *vidage* se fait généralement à des fins de *mise au point*.

07.05.02**dump**

Data that have been *dumped*.

07.05.03**change dump**

A *dump* of those *storage locations* whose contents have changed during a specified period.

07.05.04**postmortem dump**

A *dump* produced immediately after an abnormal termination of a *run*.

07.05.05**to debug**

To detect, to locate, and to eliminate *errors* in *computer programs*.

07.05.06**diagnostic program**

A *computer program* that is designed to detect, locate and describe *faults* in equipment or *errors* in *computer programs*.

07.05.07**trace program**

A *diagnostic program* that can monitor the *execution* of some or all of the *instructions* of a *computer program* and record the effects of each step.

07.05.08**checking program**

A *diagnostic program* that examines *source programs* or *data* for *errors* in syntax, semantics or conformity to other constraints.

07.05.09**to patch**

To modify an *object module*, a *load module*, or a loaded *computer program*.

NOTE — The modification is usually a temporary or expedient one.

07.05.10**assertion**

A statement that a particular condition holds at a specified point in a *program*.

NOTE — Assertions may be used for *debugging*, verification or as *comments*.

07.05.11**loop invariant**

An *assertion* that holds throughout a *loop construct*.

07.05.02**(résultat de) vidage
cliché**

Données obtenues par l'action de *vider*.

07.05.03**vidage après changement
cliché après changement**

Vidage portant sur les emplacements de *mémoire* dont le contenu a été modifié durant une période déterminée.

07.05.04**vidage d'autopsie
cliché d'autopsie**

Vidage effectué immédiatement après un arrêt anormal d'un *passage*.

07.05.05**mettre au point
débuguer**

Détecter, localiser et éliminer les *erreurs* contenues dans des *programmes d'ordinateur*.

07.05.06**programme de diagnostic**

Programme destiné à détecter, localiser et décrire des *défauts* du matériel ou des *erreurs* des *programmes d'ordinateur*.

07.05.07**programme de traçage
programme de jalonnement**

Programme de diagnostic qui peut surveiller l'*exécution* de certaines ou de toutes les *instructions* d'un *programme d'ordinateur* et enregistrer les effets de chaque étape.

07.05.08**programme de contrôle**

Programme de diagnostic conçu pour rechercher, dans des *programmes d'origine* ou des *données*, les *erreurs* de syntaxe ou de sémantique, ou pour vérifier la conformité par rapport à d'autres contraintes.

07.05.09**rapiécer**

Apporter une modification à un *module-objet*, à un *module chargeable* ou à un *programme d'ordinateur* chargé.

NOTE — Cette modification est généralement temporaire ou de circonstance.

07.05.10**assertion**

Indication qu'une condition déterminée est en vigueur en un point précis du *programme*.

NOTE — Les assertions peuvent être utilisées pour *mise au point* ou vérification, ou sous forme de *commentaires*.

07.05.11**invariant de boucle**

Assertion qui est vraie durant toute l'*exécution* d'une *boucle*.

07.05.12

precondition

An *assertion* that pertains to a point immediately preceding, in the *execution sequence*, a specified portion of a *program*.

07.05.13

postcondition

An *assertion* that pertains to a point immediately following, in the *execution sequence*, a specified portion of a *program*.

07.06 Instructions and addresses

07.06.01

instruction

statement (deprecated in this sense)

A language construct that specifies an *operation* and identifies its *operands*, if any.

07.06.02

machine instruction

An *instruction* that can be directly *executed* by a *processor* of a *computer*.

NOTE — A machine instruction is an element of a *machine language*.

07.06.03

microinstruction

An *instruction* for *operations* at a level lower than *machine instructions*.

07.06.04

instruction format

The layout of the constituent parts of an *instruction*.

07.06.05

operation part

The part of a *machine instruction* or *microinstruction* that specifies the *operation* to be performed.

07.06.06

address part

The part of a *machine instruction* or *microinstruction* that specifies one or more *addresses* of *operands*.

07.06.07

(computer) instruction code

machine code

A *code* for representing the *machine instructions* of a *computer*.

07.06.08

operation code

A *code* for representing the *operation parts* of the *machine instructions* of a *computer*.

07.05.12

précondition

postulat d'entrée

Assertion s'appliquant à un point qui, dans la *séquence d'exécution* normale, précède immédiatement une partie du *programme*.

07.05.13

postcondition

postulat de sortie

Assertion s'appliquant à un point qui, dans la *séquence d'exécution* normale, suit immédiatement une partie du *programme*.

07.06 Instructions et adresses

07.06.01

instruction

Expression qui spécifie une *opération* et, le cas échéant, en identifie les *opérandes*.

07.06.02

instruction-machine

Instruction qui peut être directement *exécutée* par un *processeur* d'un *ordinateur*.

NOTE — Une instruction-machine fait partie d'un *langage-machine*.

07.06.03

micro-instruction

Instruction s'appliquant à des *opérations* à un niveau plus élémentaire que les *instructions-machine*.

07.06.04

format d'instruction

modèle d'instruction

Disposition des différentes parties d'une *instruction*.

07.06.05

partie opérateur

partie type d'opération

Partie d'une *instruction-machine* ou d'une *micro-instruction*, qui spécifie l'*opération* à effectuer.

07.06.06

partie adresse

zone adresse

Partie d'une *instruction-machine* ou d'une *micro-instruction*, qui spécifie une ou plusieurs *adresses* d'*opérandes*.

07.06.07

code des instructions (d'un ordinateur)

Code permettant de représenter les *instructions-machine* d'un *ordinateur*.

07.06.08

code (d')opération

Code permettant de représenter les *parties opérateur* dans une *instruction-machine* d'un *ordinateur*.

07.06.09

zero address instruction

A machine instruction that has no address part.

Example: Certain instructions for a stack machine.

07.06.10

immediate instruction

An instruction that contains the value of an operand rather than its address.

NOTE — The operand is then called an immediate operand.

07.06.11

no operation instruction

no op (abbreviation)

An instruction whose execution causes the computer to proceed to the next instruction to be executed, without performing an operation.

07.06.12

privileged instruction

An instruction that can be executed only in a specific mode.

NOTE — This mode is called the privileged mode and is usually reserved for the operating system.

07.06.13

jump instruction

An instruction that specifies a jump.

07.06.14

unconditional jump instruction

A jump instruction that specifies a mandatory jump.

07.06.15

conditional jump instruction

A jump instruction that specifies a condition for the jump.

07.06.16

calling sequence

A sequence of instructions together with any associated data necessary to execute a call.

07.06.17

address

A value that identifies a location.

Examples: A register number, the address of a particular part of storage, a device address, a network address.

07.06.18

symbolic address

An identifier that represents an address.

07.06.19

base address

An address used as the origin in the calculation of addresses.

07.06.09

instruction sans adresse

Instruction-machine ne contenant pas de partie adresse.

Exemple: Certaines instructions dans une machine à pile.

07.06.10

instruction à opérande immédiat

Instruction qui contient la valeur d'un opérande plutôt que son adresse.

NOTE — L'opérande est alors appelé «opérande immédiat».

07.06.11

instruction ineffective

Instruction qui donne ordre à l'ordinateur de ne pas faire d'autre opération que de traiter l'instruction qui suit immédiatement.

07.06.12

instruction privilégiée

Instruction qui ne peut être exécutée que dans un mode particulier.

NOTE — Ce mode est dit mode privilégié; il est généralement réservé au système d'exploitation.

07.06.13

instruction de saut

Instruction qui spécifie un saut.

07.06.14

instruction de saut inconditionnel

Instruction de saut qui ne spécifie pas de condition pour le saut.

07.06.15

instruction de saut conditionnel

Instruction de saut qui spécifie une condition pour le saut.

07.06.16

séquence d'appel

Séquence d'instructions et de données associées, nécessaires pour exécuter un appel.

07.06.17

adresse

Valeur identifiant une localisation.

Exemples: Numéro d'un registre, adresse d'une partie déterminée d'une mémoire, adresse d'un appareil, adresse d'un réseau.

07.06.18

adresse symbolique

Identificateur qui représente une adresse.

07.06.19

adresse de base

Adresse utilisée comme point de départ pour le calcul des adresses.