INTERNATIONAL STANDARD

ISO/IEC 15068-2

ANSI/IEEE Std 1387.2

First edition
1999-03-15

Information technology — Portable Operating System Interface (POSIX®) system administration —

Part 2:
Software administration

Technologies de l'information — Administration du système de l'interface du système opératoire portable (POSIX®) —

Partie 2: Administration du logiciel

Reference number
ISO/IEC 15068-2:1999(E)
ANSI/IEEE
Std 1387.2, 1995 edition

AVA Documents
Document Sharing Hub

Full document details: ISO IEC 15068-2-1999

Documents
Document Sharing Hub

Full document details: ISO IEC 15068-2-1999

Information technology — Portable Operating System Interface (POSIX®) system administration — Part 2: Software administration

Copyright © 1999 by the Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street
New York, NY 10017, USA
All rights reserved.

Documents
Document Sharing Hub

Full document details: ISO IEC 15068-2-1999

Abstract: This standard is part of the POSIX® series of standards for applications and user interfaces to open systems. It defines a software packaging layout, a set of information maintained about software, and a set of utility programs to manipulate that software and information.

Keywords: data processing, open systems, operating system, packaging, portable application, POSIX®, software, system administration, utilities

POSIX is a registered trademark of the Institute of Electrical and Electronics Engineers, Inc.

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

Copyright © 1999 by the
Institute of Electrical and Electronics Engineers, Inc.
All rights reserved. Published 1996.
Printed in the United States of America.

ISBN 0-7381-1568-1

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

26 February 1999 SH94725

Documents
Document Sharing Hub

Full document details: ISO IEC 15068-2-1999

IEEE Standards documents are developed within the Technical Committees of the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE that have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of the IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, the IEEE and the members of its technical committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331

IEEE Standards documents may involve the use of patented technology. Their approval by the Institute of Electrical and Electronics Engineers does not mean that using such technology for the purpose of conforming to such standards is authorized by the patent owner. It is the obligation of the user of such technology to obtain all necessary permissions.

Documents
Document Sharing Hub

Full document details: ISO IEC 15068-2-1999

Contents

PAGE

Section 1: General 1
1.1 Scope 1
1.2 Normative References 3
1.3 Conformance 3
1.4 Test Methods 8

Section 2: Terminology and General Requirements 9
2.1 Conventions 9
2.2 Definitions 12
2.3 Dependencies on Other Standards 27

Section 3: Software Structures 29
3.1 Software_Collection 30
3.2 Distribution 32
3.3 Media 33
3.4 Installed_Software 33
3.5 Vendor 34
3.6 Software 35
3.7 Products 36
3.8 Bundles 39
3.9 Filesets 42
3.10 Subproducts 44
3.11 Software_Files 45
3.12 Files 46
3.13 Control_Files 48

Section 4: Software Administration Utilities 51
4.1 Common Definitions for Utilities 51
4.2 `swask` — Ask for user responses 90
4.3 `swconfig` — Configure software 93
4.4 `swcopy` — Copy distribution 97
4.5 `swinstall` — Install software 102
4.6 `swlist` — List software catalog 114
4.7 `swmodify` — Modify software catalog 118
4.8 `swpackage` — Package distribution 121
4.9 `swremove` — Remove software 125
4.10 `swverify` — Verify software 131

Section 5: Software Packaging Layout 137
5.1 Directory Structure 137
5.2 Software Definition File Format 141
5.3 Serial Format and Multiple Media 158

Annex A (informative) Bibliography 161

Documents
Document Sharing Hub

Full document details: ISO IEC 15068-2-1999

Annex B (informative) Rationale and Notes 165
B.1 General 165
B.2 Terminology and General Requirements 167
B.3 Software Structures 172
B.4 Software Administration Utilities 188
B.5 Software Packaging Layout 234

Annex C (informative) Sample Files 243
C.1 Defaults File 243
C.2 Product Specification File 245
C.3 Software Packaging Layout 247
C.4 INDEX File 248
C.5 INFO File 249
C.6 Control Script 250

Annex D (informative) Portability Considerations 251
D.1 User Requirements 251
D.2 Portability Capabilities 253
D.3 Profiling Considerations 254

Alphabetic Topical Index 257

FIGURES

Figure 5-1 – Example of Software Packaging Layout 138
Figure B-1 – Roles in Software Administration 168
Figure B-2 – Example of Software Structure 172
Figure B-3 – Software Object Containment 175
Figure B-4 – Software Object Inheritance 176
Figure B-5 – Fileset State Transitions (Within Distributions) 208
Figure B-6 – Fileset State Transitions (Within Installed Software) . . 209
Figure B-7 – Installation State Changes 216
Figure B-8 – Order of Install Operations 220
Figure B-9 – Order of Remove Operations 231
Figure B-10 – SDU INFO file format 237
Figure B-11 – SVR4 pkgmap file format 237

TABLES

Table 2-1 – Typographical Conventions 9
Table 3-1 – Attributes of the Software_Collection Common Class . . 31
Table 3-2 – Attributes of the Distribution Class 32
Table 3-3 – Attributes of the Media Class 33
Table 3-4 – Attributes of the Installed Software Class 34
Table 3-5 – Attributes of the Vendor Class 34
Table 3-6 – Attributes of the Software Common Class 35
Table 3-7 – Attributes of the Product Class 37

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

Table 3-8 – Attributes of the Bundle Class 40
Table 3-9 – Attributes of the Fileset Class 42
Table 3-10 – Attributes of the Subproduct Class 44
Table 3-11 – Attributes of the Software_Files Common Class 45
Table 3-12 – Attributes of the File Class 47
Table 3-13 – Attributes of the Control File Class 49
Table 4-1 – Software_spec Version Identifiers 56
Table 4-2 – Script Return Codes 73
Table 4-3 – Event Status 74
Table 4-4 – General Error Events 75
Table 4-5 – Session Events 76
Table 4-6 – Analysis Phase Events 78
Table 4-7 – Execution Phase Events 83
Table 4-8 – Return Codes 89
Table 4-9 – Default Levels 116
Table 5-1 – File Attributes for INFO File 153
Table B-1 – Possible Attributes of a Host Class 177
Table B-2 – Mapping from Software to DMTF Component ID 180
Table B-3 – Comparison of Some Existing Practices 192
Table B-4 – Comparison of Software Administration Packages . . . 217
Table B-5 – Comparison of Existing Practice for Software Packaging . 228
Table B-6 – Packaging Layout Comparisons 236
Table D-1 – 1387.2 Portability Capability Summary 253

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

International Standard ISO/IEC 15068-2:1999(E)

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. 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 voting. Publication as an International Standard requires approval by at least 75% of the national bodies casting a vote.

International Standard ISO/IEC 15068-2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology,* Subcommittee SC22, *Programming languages, their environments and system software interfaces.*

ISO/IEC 15068 consists of the following parts, under the general title *Information technology —Portable Operating System Interface (POSIX):*

— *Part 1:* (reserved for future use)

— *Part 2: Software administration*

— *Part 3: User and group account administration*

— *Part 4: Print administration*

Annexes A to D of this part of ISO/IEC 15068 are for information only.

Documents
Document Sharing Hub

Full document details: ISO IEC 15068-2-1999

Introduction

(This Introduction is not a normative part of ISO/IEC 15068-2 Information technology — Portable Operating System Interface (POSIX) system administration — Part 2: Software administration, but is included for information only.)

System administration utilities vary widely between vendors, being an area where there are currently no formal standards that have proved to be significant in practice. This makes the task of system administration difficult. The objective of this part of ISO/IEC 15068 is to address this problem for software administration, a specific area of system administration, and to contribute to the overall solution of administering computing environments, both stand-alone and distributed.

In pursuit of this goal, this part of ISO/IEC 15068 defines a software packaging layout, a set of information maintained about software, and a set of utility programs to manipulate that software and information. These definitions provide the flexibility necessary for system administrators to enforce policies suitable to their environments.

Organization of the Standard

The standard is divided into the following sections:

(1) General

(2) Terminology and General Requirements

(3) Software Structures

(4) Software Administration Utilities

(5) Software Packaging Layout

Also included are the following annexes:

— Bibliography (Annex A)

— Rationale and Notes (Annex B)

— Sample Files (Annex C)

— Portability Considerations (Annex D)

This introduction and the annexes are not considered a normative part of the standard.

Conformance Measurement

In publishing this part of ISO/IEC 15068, both IEEE and the POSIX.7.2 developers simply intend to provide a yardstick against which various operating system implementations may be measured for conformance. It is not the intent of either the IEEE or POSIX.7.2 developers to measure or rate any products or to reward or sanction the product of any vendor as standard by these or any other means. The responsibility for determining the degree of conformance or lack thereof with this part of ISO/IEC 15068 rests solely with the individual evaluating the product claiming to be in conformance with the standard.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

Base Documents

Much of the original text came to the developers of this part of ISO/IEC 15068 from UNIX System Laboratories (the `pkg*` utilities) and Hewlett-Packard (HP Software Distribution Utilities). For further details and comparisons of various existing practices, see B.4 and B.4.5.

Extensions and Supplements to This Standard

Activities to extend this standard to address additional requirements are in progress and similar efforts can be anticipated in the future. This is an outline of how these extensions will be incorporated and how users of this document can keep track of that status.

Extensions are approved as supplements to this document following the IEEE standards procedures and eventually as International Organization for Standardization/International Electrotechnical Committee (ISO/IEC) standards.

Approved supplements are published separately and distributed with orders from the IEEE for this document until the full document is reprinted and such supplements are incorporated in their proper positions.

If there are any questions about the completeness of your version, you may contact the IEEE Computer Society, (202) 371-0101, or the IEEE Standards Office, (908) 562-3800, to determine what supplements have been published.

Supplements may contain either required functions or optional facilities. Supplements may add additional conformance requirements (see 1.3, which defines new classes of conforming systems or applications).

It is undesirable (but perhaps unavoidable) for supplements to change the functionality of the already defined facilities.

If you are interested in participating in addressing IEEE 1387 issues and developing the IEEE P1387 standards, please send your name, address, and phone number to the Secretary, IEEE Standards Board, P.O. Box 1331, 445 Hoes Lane, Piscataway, NJ 08844-1331, USA.

When writing, ask to have your letter forwarded to the chairperson or appropriate reviewer/developer of IEEE Std 1387.2-1995.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

ISO/IEC 15068-2 was prepared by the IEEE P1387.2 Working Group, sponsored by the Portable Applications Standards Committee of the IEEE Computer Society. At the time this part of ISO/IEC 15068 was approved, the membership of the IEEE P1387.2 Working Group was as follows:

Portable Applications Standards Committee

Chair:	Lowell Johnson
Vice-Chair:	Charles Severance
Functional Chairs:	Andy Bihain
	Jon Spencer
	Andrew Josey
	Jay Ashford
	Barry Needham
Treasurer:	Peter Smith
Secretary:	Charles Severance

IEEE P1387 Working Group Officials

Chair:	Martin Kirk
	Steve Carter (1988-1990)
Vice-Chair:	Jay Ashford
	David Hinnant (1988-1991)
Technical Editor:	Matt Wicks (1991-1993)
	Robert Robillard (1991-1993)
	Shoshana O'Brien (1989-1991)
	Bob Baumann (1988-1990)
Secretary:	Bernard Kinsler (1990-1995)
	John Pokladnik (1990)
	Mark Colburn (1989)

IEEE P1387.2 Working Group Officials

Chair:	Jay Ashford
Technical Editors:	Jay Ashford, George Williams
	Matt Wicks (1991-1993)

IEEE P1387.2 Technical Reviewers

Jay Ashford	Jerry Rubin	George Williams
	Matt Wicks	

IEEE P1387.2 Working Group

Richard Alexander	Louis Imershein	Ken Nicholson
Barrie Archer	John Jobs	Per Pedersen
Jay Ashford	Jim Johnson	Daryl Roberts
Shane Claussen	Judy Kale	Helmut Roth
Jim Darling	Martin Kirk	Jerry Rubin
Frances Dodson	Esti Koen	Nigel Titley
Frank Dogil	Steve Lamotte	Stephe Walli
Janet Frazer	Sean Landis	Matthew Wicks
Jay Goldberg	Fu-Tin Man	George Williams
Michael Gutmann	Norbert Marrek	Neil Winton
Steve Howell	Laura Micks	Jane Zysk
David Humphreys	Tom Murphy	

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

The following persons provided valuable input during the balloting period:

Francesco Borgna
Theodore Collins
Cheng Hu
Shane P. McCarron
Brenda Parsons
Dieter Putatzki
Walter Wong
Mike Ryan
Larry Spieler
Marc J. Stephenson

The following persons were on the balloting committee:

Barrie Archer
Jay Ashford
Jason Behm
Michael E. Browne
Dana Carson
Shane Claussen
Frances Dodson
Ron Elliott
Michael E. Falck
David Fiander
Dan Geer
Michel Glen
Dave Grindeland
Geoff Hall
Barry Hedquist
Joseph Hungate
Louis Imershein
Hal Jespersen
Judy S. Kerner
Lawrence Kilgallen
Martin J. Kirk
Esti Koen
George Kriger
Thomas M. Kurihara
Sean Landis
Jim Moore
John S. Morris
Mo Oloumi
Paul Rabin
David Radford
Rick Roelling
Frank Rone
Jerrold Rubin
James G. Tanner
Mark-Rene Uchida
Matthew Wicks
George Williams
Walter Wong
Oren Yuen

When the IEEE Standards Board approved this standard on June 14, 1995, it had the following membership:

E. G. "Al" Kiener, *Chair* **Donald C. Loughry,** *Vice Chair*
Andrew G. Salem, *Secretary*

Gilles A. Baril
Clyde R. Camp
Joseph A. Cannatelli
Stephen L. Diamond
Harold E. Epstein
Donald C. Fleckenstein
Jay Forster*
Donald N. Heirman
Richard J. Holleman
Jim Isaak
Ben C. Johnson
Sonny Kasturi
Lorraine C. Kevra
Ivor N. Knight
Joseph L. Koepfinger*
D. N. "Jim" Logothetis
L. Bruce McClung
Marco W. Migliaro
Mary Lou Padgett
John W. Pope
Arthur K. Reilly
Gary S. Robinson
Ingo Rusch
Chee Kiow Tan
Leonard L. Tripp

*Member Emeritus

Also included are the following nonvoting IEEE Standards Board liaisons:

Satish K. Aggarwal
Richard B. Engelman
Robert E. Hebner
Chester C. Taylor

Rochelle L. Stern
IEEE Standards Project Editor

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

Documents
Document Sharing Hub

Full document details: ISO IEC 15068-2-1999

Information technology — Portable Operating System Interface (POSIX) system administration — Part 2: Software administration

Section 1: General

1.1 Scope

This part of ISO/IEC 15068 defines a software packaging layout and utilities that operate on that packaging layout as well as software installed from that packaging layout. The scope of this part of ISO/IEC 15068 is administration of software across distributed systems. This administration includes, but is not limited to, packaging of software for distribution, distribution of software to systems, installation and configuration of software on systems, and removal of software from systems.

This part of ISO/IEC 15068 is motivated by many factors, including a desire by system administrators and software suppliers to have a common way of installing and removing software. To meet the needs of these groups, this part of ISO/IEC 15068 consists of several components, listed below. The readers of this part of ISO/IEC 15068 include system administrators, suppliers of software that implement this part of ISO/IEC 15068, and suppliers of software that use implementations of this part of ISO/IEC 15068. Readers in each of these categories may find their attention drawn to different sections.

The key components are listed below.

Software structures

This part of ISO/IEC 15068 defines a hierarchical set of structures used to define software. Information is kept about the software based on these structure definitions. The structure definitions apply both to installed software and to software prepared for installation

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

but not yet installed.

Software packaging layout

This part of ISO/IEC 15068 defines the organization of software on a distribution medium, the information held about that software, and the way in which such information is represented. This enables both portability of software distributions across systems of different architecture, and the use of different media to distribute software (including both file system and serial image forms).

Information kept about software

This part of ISO/IEC 15068 defines the information that is held about software, both installed software and distributions. This provides a consistent view of software, even when that software is provided from various sources. The way in which the information is held is undefined within this part of ISO/IEC 15068.

Utilities to administer software

This part of ISO/IEC 15068 defines a utility to convert software into the packaging layout, known as a distribution. This part of ISO/IEC 15068 also contains utilities to examine the information in a distribution, copy software from one distribution to another, install software from a distribution, remove software from a distribution, and verify the integrity of a distribution. There are also utilities for configuring installed software, verifying the integrity of installed software, examining and modifying the information held about installed software, and for removing installed software from a system. This provides administrators with a consistent method of dealing with software across all conforming systems.

Distributed software administration

This part of ISO/IEC 15068 defines the concepts, and the utility syntax and behaviors, for managing software in a distributed environment. This includes the concept of different software administration roles (developer, packager, manager, source, target, and client). Different utilities involve different roles, and different roles may be distributed across multiple systems within a single command execution.

This part of ISO/IEC 15068 is based upon the knowledge of, and documentation for, existing programs that assume an interface and architecture similar to that described by POSIX.1 {2}[1)] and POSIX.2 {3}. Any questions regarding the definition of terms or the semantics of an underlying concept should be referred to POSIX.1 {2} and POSIX.2 {3}. This part of ISO/IEC 15068 does not require the use of any specific programming language and, in particular, does not require the use of the C language. This part of ISO/IEC 15068 is based upon the knowledge of, and documentation for, existing programs that utilize C-language interfaces. Any questions regarding the definition of terms or the semantics of an underlying

1) The numbers in curly brackets correspond to those of the references in 1.2. When preceded by a "B," the numbers correspond to those of the bibliography in Annex A.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

concept in this language should be referred to C Standard {B13}.

1.2 Normative References

The following standards contain provisions that, through references in this text, constitute provisions of this part of ISO/IEC 15068. 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 15068 are encouraged to investigate the possibility of applying the most recent editions of the standards listed below. Members of IEC and ISO maintain registers of currently valid International Standards.

{1} ISO/IEC 646: 1991, *Information technology—ISO 7-bit coded character set for information interchange* (International Reference Version).[2)]

{2} ISO/IEC 9945-1: 1996 (ANSI/IEEE Std 1003.1-1996), *Information technology — Portable Operating System Interface (POSIX®) — Part 1: System Application Program Interface (API) [C Language].* [3)]

{3} ISO/IEC 9945-2: 1993 (ANSI/IEEE Std 1003.2-1992), *Information technology — Portable Operating System Interface (POSIX®) — Part 2: Shell and Utilities.*

{4} ISO/IEC 10646-1: 1993, *Information technology—Universal Multiple-Octet Coded Character Set (UCS)—Part 1: Architecture and Basic Multilingual Plane.*

1.3 Conformance

1.3.1 Implementation Conformance

1.3.1.1 Conforming POSIX.7.2 Implementation

A Conforming POSIX.7.2 Implementation, also known as a "conforming implementation," shall meet all the following criteria:

(1) The system shall support all interfaces defined within this part of ISO/IEC 15068. These interfaces shall support all the functional behavior described herein. The interfaces covered by this definition of conformance include, but are not limited to, utilities and their options and extended options, the behavior of the utilities, including the generation of events; events; structures; attributes and their values; and file formats.

---

2) ISO/IEC documents can be obtained from the ISO Central Secretariat, Case Postale 56, 1 rue de Varembé, CH-1211, Genève 20, Switzerland/Suisse.

3) IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

(2) The system may provide additional or enhanced utilities, functions, or facilities not required by this part of ISO/IEC 15068. Nonstandard extensions should be identified as such in the system documentation. Nonstandard extensions should conform to 2.10.2, of POSIX.2 {3}. Nonstandard extensions, when used, may change the behavior of utilities, functions, or facilities defined by this part of ISO/IEC 15068. In such cases, the conformance document for the implementation (see 2.2.1.3) shall define an execution environment (i.e., shall provide general operating instructions) in which a Strictly Conforming POSIX.7.2 Distribution may be operated upon and yield the behavior specified by this part of ISO/IEC 15068. In no case shall such an environment require modification of a Strictly Conforming POSIX.7.2 Distribution.

An implementation shall be a Conforming POSIX.7.2 Implementation if all the criteria for such are met with the exception of those features that depend on the existence of conforming implementations of either POSIX.1 {2} or POSIX.2 {3}. In this case, the conformance document for the Conforming POSIX.7.2 Implementation shall describe the behavior of the implementation of all features of the implementation, or of a Strictly Conforming POSIX.7.2 Distribution, that depend on the function of POSIX.1 {2} or POSIX.2 {3}. See 2.3.1 and 2.3.2.

1.3.1.2 Limited Conformance POSIX.7.2 Implementation

A Limited Conformance POSIX.7.2 Implementation shall meet all of the criteria established for a Conforming POSIX.7.2 Implementation (see 1.3.1.1) with the following exception:

— For the value of **HOST** in specifications of sources and targets (see 4.1.4.2), the system may support only the local machine. While this type of limited conformance removes support for remote operations, the syntax of all utilities and files shall remain identical to that required for Conforming POSIX.7.2 Implementations. The way in which this limitation is imposed by the implementation shall be implementation defined.

1.3.1.3 Documentation

A conformance document with the following information shall be available for an implementation claiming conformance to this part of ISO/IEC 15068. The conformance document shall have the same structure as this part of ISO/IEC 15068, with the information presented in the appropriately numbered sections. Sections that consist solely of subordinate section titles, with no other information, are not required.

The conformance document shall not contain information about extended facilities or capabilities outside the scope of this part of ISO/IEC 15068, unless those extensions affect the behavior of a Strictly Conforming POSIX.7.2 Distribution; in such cases, the documentation required by the previous subclause shall be included.

The conformance document shall contain a statement that indicates the full name, number, and date of this part of ISO/IEC 15068 that applies. The conformance document may also list software standards approved by ISO/IEC or any ISO/IEC member body that are available for use by a Conforming POSIX.7.2 Implementation or by a Conforming POSIX.7.2 Distribution. Applicable characteristics where

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

documentation is required by one of these standards, or by standards of government bodies, may also be included.

The conformance document shall describe the behavior of the implementation for all implementation-defined features defined in this part of ISO/IEC 15068. This requirement shall be met by listing these features and providing either a specific reference to the system documentation or full syntax and semantics of these features. When the value or behavior in the implementation is designed to be variable or customizable on each instantiation of the system, the implementation provider shall document the nature and permissible ranges of this variation. When information required by this part of ISO/IEC 15068 is related to the underlying operating system and is already available in the POSIX.1 {2} or POSIX.2 {3} conformance document, the implementation need not duplicate this information in the conformance document for this part of ISO/IEC 15068, but may provide a cross-reference for this purpose.

The conformance document shall indicate whether the implementation is based on an underlying operating system that is fully conforming to both POSIX.1 {2} and POSIX.2 {3}.

If the implementation is not based on a conforming implementation of POSIX.1 {2}, then the conformance document shall describe the behavior of the implementation for all features of a Strictly Conforming POSIX.7.2 Distribution that depend on the function of POSIX.1 {2}. See 2.3.1.

If the implementation is not based on a fully conforming implementation of POSIX.2 {3}, then the conformance document shall describe the behavior of the implementation for all features of a Strictly Conforming POSIX.7.2 Distribution that depend on the function of POSIX.2 {3}. This dependency includes all of the utilities of POSIX.2 {3}, including the shell. See 2.3.2.

The conformance document may specify the behavior of the implementation for those features where this part of ISO/IEC 15068 states that implementations may vary or where features are identified as undefined or unspecified.

No specifications other than those described in this subclause (1.3.1.3) shall be present in the conformance document.

The phrase "shall be documented" in this part of ISO/IEC 15068 means that documentation of the feature shall appear in the conformance document, as described previously, unless the system documentation is explicitly mentioned.

The system documentation should also contain the information found in the conformance document.

1.3.1.4 Conforming Implementation Options

Additional utility options, distribution formats, or software structure attributes and values, may be provided in other related standards or in future revisions to this part of ISO/IEC 15068, without requiring this part of ISO/IEC 15068 to be updated.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

1.3.2 Distribution Conformance

All distributions claiming conformance to this part of ISO/IEC 15068 fall within one of the categories in the following subclauses.

1.3.2.1 Strictly Conforming POSIX.7.2 Distribution

A Strictly Conforming POSIX.7.2 Distribution is a distribution that requires only the facilities described in this part of ISO/IEC 15068 (including any required facilities of the underlying operating system; see 2.3). Such a distribution

(1) Shall contain only those files and directories defined in the Software Packaging Layout (see Section 5)

(2) Shall contain only those software structures and attributes within those structures described in this part of ISO/IEC 15068

(3) Shall use only the values of software structure attributes defined as valid within described in this part of ISO/IEC 15068

(4) Shall use only the POSIX.2 {3} shell, **sh**, to invoke control files

(5) Shall not use facilities, structures, attributes, or values designated as *obsolescent*

Within this part of ISO/IEC 15068, any restrictions placed upon a Conforming POSIX.7.2 Distribution also shall restrict a Strictly Conforming POSIX.7.2 Distribution.

1.3.2.2 Conforming POSIX.7.2 Distribution

A Conforming POSIX.7.2 Distribution is a distribution that uses only the facilities described in this part of ISO/IEC 15068 and implied facilities of the underlying operating system. See 2.3. Such a distribution shall use only the POSIX.2 {3} shell, **sh**, to invoke control files. In addition to the software structures, and attributes within those structures, allowable in a Strictly Conforming POSIX.7.2 Distribution, a Conforming POSIX.7.2 Distribution can also contain:

(1) Additional attributes in the manner supported by this part of ISO/IEC 15068.

(2) Additional files within the distribution, except where this part of ISO/IEC 15068 prohibits such files.

The term Conforming POSIX.7.2 Distribution is used to describe either of the two following distribution types.

1.3.2.2.1 ISO/IEC Conforming POSIX.7.2 Distribution

An ISO/IEC Conforming POSIX.7.2 Distribution shall also include a statement of conformance that documents all other ISO/IEC standards used.

1.3.2.2.2 <National Body> Conforming POSIX.7.2 Distribution

A <National Body> Conforming POSIX.7.2 Distribution differs from an ISO/IEC Conforming POSIX.7.2 Distribution in that it also may use specific standards of a single ISO/IEC member body referred to here as "<*National Body*>." Such a

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

distribution shall include a statement of conformance that documents all other <*National Body*> standards used.

1.3.2.3 Conforming POSIX.7.2 Distribution Using Extensions

A Conforming POSIX.7.2 Distribution Using Extensions is a distribution that differs from a Conforming POSIX.7.2 Distribution only in that it either requires behavior of an implementation other than that which is specified by this part of ISO/IEC 15068, or requires invocation of control files with other than the POSIX.2 {3} shell, or both.

NOTE: One example of non-standard behavior is that of requiring implementations to understand, and take action based upon, the values of various vendor-defined attributes. Another example of non-standard behavior is the ability to properly handle files with names drawn from other than the portable filename character set. Vendors creating such distributions are encouraged to transform filenames using UTF-8 {4}.

In addition to the documentation required of a Conforming POSIX.7.2 Distribution, such a distribution

(1) Shall fully document its requirements for these extended facilities, whether required within the software packaging layout, within the utilities defined in this part of ISO/IEC 15068, or both

(2) Shall document whether it is possible to convert this distribution into a Conforming POSIX.7.2 Distribution, and what those steps are

(3) Shall document the extent to which the software contained in the distribution can be managed by the utilities in accordance with this part of ISO/IEC 15068

(4) Shall document the requirement for any interpreter other than the POSIX.2 {3} Shell

A Conforming POSIX.7.2 Distribution Using Extensions shall be either an ISO/IEC Conforming POSIX.7.2 Distribution Using Extensions, or a <National Body> Conforming POSIX.7.2 Distribution Using Extensions. See 1.3.2.2.1 and 1.3.2.2.2.

1.3.2.4 Documentation

A conformance document with information required by 1.3.2 shall be available for a distribution claiming conformance to this part of ISO/IEC 15068.

The conformance document shall contain a statement that indicates the full name, number, and date of this part of ISO/IEC 15068 that applies. The conformance document may also list software standards approved by ISO/IEC or any ISO/IEC member body that are available for use by a Conforming POSIX.7.2 Implementation or by a Conforming POSIX.7.2 Distribution. Applicable characteristics where documentation is required by one of these standards, or by standards of government bodies, may also be included.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

1.4 Test Methods

There are no specific test methods for this part of ISO/IEC 15068.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

Section 2: Terminology and General Requirements

2.1 Conventions

2.1.1 Editorial Conventions

This part of ISO/IEC 15068 uses the following editorial and typographical conventions. A summary of typographical conventions is shown in Table 2-1.

Table 2-1 – Typographical Conventions

Reference	Example
Attribute of a Structure (object)	*filesets* or *product.filesets*
C-Language Data Type	*long*
C-Language Function	*system*()
Cross Reference: Annex	Annex A
Cross Reference: Clause	2.3
Cross Reference: Other Standard	ISO/IEC 9999-1 {*n*}
Cross Reference: Section	Section 2
Cross Reference: Subclause	2.3.4, 2.3.4.5, 2.3.4.5.6
Defined Term	(see text)
Definition Citation	[POSIX.1 {2}]
Environment Variable	**PATH**
Error Number	[EINTR]
Event Status: Event Code	SW_ERROR: SW_IO_ERROR
Example Input	`echo Hello, World`
Example Output	`Hello, World`
Figure Reference	Figure 7-1
File Name	`/tmp`
Keyword	`file_permissions`
Parameter	*<directory pathname>*
Script Types	`preinstall`
Special Character	`<newline>`
Symbolic Constant, Limit	{_POSIX_VDISABLE}, {LINE_MAX}
Table Reference	Table 6-1
Utility Extended Option	*reinstall*
Utility Extended Option with Value	*reinstall=true*
Utility Name	`awk`
Utility Operand	*file_name*
Utility Option	`-c`
Utility Option with Option-Argument	`-w` *width*
Value of an Attribute	`true`

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

The **Bold** font is used to show brackets that denote optional arguments in a utility synopsis, as in

cut [-c *list***] [***file_name***]**

These brackets shall not be used by the application unless they are specifically mentioned as literal input characters by the utility description.

There are symbols enclosed in angle brackets (**< >**):

Parameters — Parameters, also called *metavariables*, are in italics, such as <*directory pathname*>. The entire symbol, including the brackets, is meant to be replaced by the value of the symbol described within the brackets.

In some examples, the **Bold Courier** font is used to indicate the output of the system that resulted from some user input, shown in **Courier**.

Defined terms are shown in three styles, depending on context:

(1) Terms defined in 2.2.1, and 2.2.2 are expressed as subclause titles. Alternative forms of the terms appear in [brackets]. At the conclusion of this type of definition, the designation of a standard within [brackets] indicates that the text of the definition was copied from that standard, with only editorial changes, if any.

(2) The initial appearances of other terms, applying to a limited portion of the text, are in *italics*.

(3) Subsequent appearances of the term are in the Roman font.

Symbolic constants are shown in the following two styles — those within curly brackets are intended to call the attention of the reader to values in **<limits.h>** and **<unistd.h>**; those without braces are usually defined by one or a few related functions. There is no semantic difference between these two forms of presentation.

Events are shown within text in uppercase form and enclosed in parentheses. This is done to call the attention of the reader to the values described in 4.1.6.2.

Filenames and pathnames are shown in **Courier**. When a pathname is shown starting with **$HOME/**, this indicates the remaining components of the pathname are to be related to the directory named by the user's **HOME** environment variable.

The style selected for some of the special characters, such as **<newline>**, matches the form of the input given to the **localedef** utility (see 2.5.2 of POSIX.2 {3}). Generally, the characters selected for this special treatment are those that are not visually distinct, such as the control characters **<tab>** or **<newline>**.

Literal characters and strings used as input or output are shown in various ways, depending on context as follows:

%, **begin** — When no confusion would result, the character or string is rendered in the Courier font and used directly in the text.

Documents Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

`'c'` In some cases a character is enclosed in single-quote characters, similar to a character constant used by several programming languages. Unless otherwise noted, the quotes shall not be used as input or output.

`"string"` In some cases, a string is enclosed in double-quote characters similar to a string constant used by several programming languages. Unless otherwise noted, the quotes shall not be used as input or output.

Defined names that are usually in lowercase, particularly function names, are never used at the beginning of a sentence or anywhere else that English usage would require them to be capitalized.

Parenthetical expressions within normative text also contain normative information. The general typographic hierarchy of parenthetical expressions is:

{ [( )] }

The square brackets are most frequently used to enclose a parenthetical expression that contains a function name [such as *waitpid*()] with its built-in parentheses.

In some cases, tabular information is presented inline; in others it is presented in a separately labeled table. This arrangement was employed purely for ease of reference and there is no normative difference between these two cases.

Annexes marked as *normative* are parts of the standard that pose requirements, exactly the same as the numbered sections, but have been moved to near the end of the document for clarity of exposition. *Informative* annexes are for information only and pose no requirements. All material preceding page 1 of the document (the "front matter") and the two indexes at the end are also only informative.

NOTES that appear in a smaller point size have one of two different meanings as follows, depending on their location:

— When they are within the normal text of the document, they are the same as footnotes — informative, posing no requirements on implementations, distributions, or applications.

— When they are attached to tables or figures, they are normative, and can include requirements.

Text marked as examples (including the use of "e.g.") is for information only.

The table in this section refers to constructs from the C language. This part of ISO/IEC 15068 itself has no requirement for the use of the C language or any other programming language in particular. However, this part of ISO/IEC 15068 has been developed with the use of concepts from POSIX.1 {2} and POSIX.2 {3}, and those standards do make use of C-language constructs. The use of C-language constructs within this part of ISO/IEC 15068 is simply to provide reference to the appropriate parts of POSIX.1 {2} and POSIX.2 {3}. As far as this part of ISO/IEC 15068 is concerned, use of equivalent function from other languages is acceptable.

In some cases certain characters are interpreted as special characters by the shell. In the normative portions of this part of ISO/IEC 15068, these characters are shown without escape characters or quoting (see 3.2 in POSIX.2 {3}). In all examples, however, quoting has been used, showing how sample commands

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

(utility names combined with arguments) could be passed correctly to a shell (see `sh` in 4.56 of POSIX.2 {3}), or as a string to the *system*() function.

The typographical conventions listed here are for ease of reading only. Editorial inconsistencies in the use of typography are unintentional and have no normative meaning in this part of ISO/IEC 15068.

2.1.2 Grammar Conventions

Portions of this part of ISO/IEC 15068 are expressed in terms of a special grammar notation. It is used to portray the complex syntax of certain program input. The grammar is based on the syntax used by the `yacc` utility (see A.3 in POSIX.2 {3}). However, it does not represent fully functional `yacc` input suitable for program use. The lexical processing and all semantic requirements are described only in textual form. The grammar is not based on source used in any traditional implementation and has not been tested with the semantic code that would normally be required to accompany it. Furthermore, there is no implication that the partial `yacc` code presented represents the most efficient, or only, means of supporting the complex syntax within the utility. Implementations may use other programming languages or algorithms, as long as the syntax supported is the same as that represented by the grammar.

The following typographical conventions are used in the grammar; they have no significance except to aid in reading.

— The identifiers for the reserved words of the language are shown with a leading capital letter. These are terminals in the grammar. Examples: `While`, `Case`.

— The identifiers for terminals in the grammar are all named with uppercase letters and underscores. Examples: `NEWLINE`, `ASSIGN_OP`, `NAME`.

— The identifiers for nonterminals are all lowercase.

2.2 Definitions

2.2.1 Terminology

For the purposes of this part of ISO/IEC 15068, the following definitions apply:

2.2.1.1 application: Includes both humans and executable programs that use implementations of this part of ISO/IEC 15068. Such executable programs can include control files within a distribution.

2.2.1.2 can: An indication of a permissible optional feature or behavior available to a distribution or to someone using a utility defined in this part of ISO/IEC 15068; the implementation shall support such features or behaviors as mandatory requirements.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.2.1.3 conformance document: A document provided by an implementor that contains implementation details as described in 1.3.1.3.

NOTE: See 2.2.1.3 of POSIX.2 {3}.

2.2.1.4 implementation: An object providing, to distributions and users, the services defined by this part of ISO/IEC 15068.

The word *implementation* is to be interpreted to mean that object, after it has been modified in accordance with the manufacturer's instructions to:

— Configure it for conformance with this part of ISO/IEC 15068;

— Select some of the various optional facilities described by this part of ISO/IEC 15068, through customization by local system administrators or operators.

An exception to this meaning occurs when discussing conformance documentation or using the term *implementation defined*.

NOTE: See 2.2.1.5 and 1.3.1.3.

2.2.1.5 implementation defined: An indication that the implementation provider shall define and document the requirements for correct program constructs and correct data of a value or behavior.

When the value or behavior in the implementation is designed to be variable or customizable on each instantiation of the system, the implementation provider shall document the nature and permissible ranges of this variation.

NOTE: See 1.3.1.3. See also 2.2.1.5 of POSIX.2 {3}.

2.2.1.6 may: An indication of an optional feature or behavior of the implementation that is not required by this part of ISO/IEC 15068, although there is no prohibition against providing it.

A Strictly Conforming POSIX.7.2 Distribution is permitted to use such features, but shall not rely on the implementation's actions in such cases. To avoid ambiguity, the reverse sense of *may* is not expressed as *may not*, but as *need not*.

2.2.1.7 obsolescent: An indication that a certain feature may be considered for withdrawal in future revisions of this part of ISO/IEC 15068.

NOTE: See 2.2.1.7 of POSIX.2 {3}.

2.2.1.8 shall: An indication of a requirement on the implementation or on Strictly Conforming POSIX.7.2 Distributions, where appropriate.

2.2.1.9 should:

(1) With respect to implementations, an indication of an implementation recommendation, but not a requirement.

(2) With respect to distributions, an indication of a recommended practice for distributions, and a requirement for Strictly Conforming POSIX.7.2

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

Distributions.

2.2.1.10 system documentation: All documentation provided with an implementation, except the conformance document.

Electronically distributed documents for an implementation are considered part of the system documentation.

NOTE: See 2.2.1.10 of POSIX.2 {3}.

2.2.1.11 undefined: An indication that this part of ISO/IEC 15068 imposes no portability requirements on applications for erroneous program construction, erroneous data, or use of an indeterminate value.

Implementations (or other standards) may specify the result of using that value or causing that behavior. An application using such behaviors is using extensions, as defined in 1.3.2.3.

NOTE: See 2.2.1.11 of POSIX.2 {3}.

2.2.1.12 unspecified: An indication that this part of ISO/IEC 15068 imposes no portability requirements on applications for a correct program construction or correct data.

Implementations (or other standards) may specify the result of using that value or causing that behavior. An application requiring a specific behavior, rather than tolerating any behavior when using that functionality, is using extensions, as defined in 1.3.2.3.

NOTE: See 2.2.1.12 of POSIX.2 {3}.

2.2.2 General Terms

For the purposes of this part of ISO/IEC 15068, the following definitions apply:

2.2.2.1 absolute path: If the underlying system is based upon a conforming implementation of POSIX.1 {2}; then a pathname that begins with **/**; otherwise, *absolute path* is implementation defined.

2.2.2.2 alternate root directory: A pathname other than **/** for managing installed software.

2.2.2.3 analysis phase: The steps a software administration utility performs, before modifying the target, while attempting to ensure that the execution of operations on the target will succeed.

2.2.2.4 attribute: A component of an object, possessing a name and one or more values.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.2.2.5 autorecovery: The process of restoring installed software to the state it was in prior to the invocation, and subsequent failure during execution, of the `swinstall` utility.

2.2.2.6 autoselect: The automatic selection, within a utility, of software beyond that directly specified by the user in order to meet the dependencies of the user-specified software.

2.2.2.7 bundle: A software object, which is a grouping of other software objects, such as all or parts of other bundles and products.

NOTE: See 3.8.

2.2.2.8 catalog: The metadata describing all the software objects that are a part of a single software_collection (distribution or installed_software object).

Catalogs exist both in distributions and for installed software, although storage of catalogs for installed software is undefined within this part of ISO/IEC 15068.

A catalog in a distribution shall always use the exported catalog structure, since it is required to be stored in a portable or exported catalog structure. A catalog for installed software shall use the exported catalog structure when information is listed with `swlist -v`.

NOTE: See Section 5.

2.2.2.9 class: Describes the structure and attributes of each level of the software hierarchy that is used to organize and manage software files.

2.2.2.10 client role: The location where the software is actually executed or used (as opposed to the target where it is actually installed).

The configuration of software is performed by this role.

2.2.2.11 command line interface: A means of invoking utilities by issuing commands from within a POSIX.2 {3} shell, implying that neither graphics nor windows are required.

2.2.2.12 common class: Defines those aspects of different software objects that are the same.

The common classes for this part of ISO/IEC 15068 are software_collections, software, and software_files. The names of these classes are also used to generically describe any object that shares that common class.

2.2.2.13 compressed file: A file that has been transformed in a manner intended to reduce its size without loss of information.

2.2.2.14 containment: A relationship between two objects such that one is said to belong to, or form part of, the other.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

All objects except software_collection objects shall be contained within exactly one object. The containment of software_collection objects is undefined within this part of ISO/IEC 15068.

2.2.2.15 control directory: The directory below which the control_files for filesets and products are stored within exported catalogs for distributions and installed software.

2.2.2.16 control_files: The control scripts executed by the utilities, the INFO file describing the files in a fileset, and other files associated with a software object.

2.2.2.17 control script: A control_file associated with a software object that is executed by the software administration utilities.

2.2.2.18 corequisite: The specification in a software object such that another software object shall be installed, in conjunction with the installation of the first, and configured in conjunction with the configuration of the first.

2.2.2.19 decimal character string: A sequence of characters from the set of decimal digits the first of which shall not be the digit zero.

Decimal character strings shall consist only of the following characters:

`0 1 2 3 4 5 6 7 8 9`

Within software definition files of exported catalogs, all such strings shall be encoded using IRV {1}.

2.2.2.20 default option: The value for an extended option as defined in a defaults file.

NOTE: See 2.2.2.21 and 2.2.2.35.

2.2.2.21 defaults file: A system-specific or user-specific file that contains the default values for extended options used by the software administration utilities.

2.2.2.22 dependency: A software object that is a prerequisite, corequisite or exrequisite for a software object defining a `dependency_spec`.

A dependency is the object upon which another object depends.

2.2.2.23 dependency_spec: A `software_spec` that describes a dependency.

NOTE: See 4.1.4.1.

2.2.2.24 dependent: A software object that specifies a prerequisite, corequisite or exrequisite on another software object.

2.2.2.25 developer role: Where software is developed, tested, and maintained.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

This role is outside the scope of this part of ISO/IEC 15068.

2.2.2.26 directory medium: A medium that contains a distribution in a POSIX.1 {2} hierarchical file system format.

NOTE: An example of this is a distribution contained in a POSIX.1 {2} file system format on a CD-ROM.

2.2.2.27 distribution: A software_collection containing software in the software packaging layout.

2.2.2.28 distribution catalog: The catalog of metadata for a distribution software_collection.

Unlike a catalog for an installed_software object, a distribution catalog is stored in a particular exported catalog structure that is part of the software packaging layout.

2.2.2.29 distribution path: The pathname below which the catalog describing the distribution is located.

If the distribution is on a single medium, all software for it is located below this path.

2.2.2.30 downdate: Installation of software with a revision older than that of the software currently installed in the same location.

This is also referred to as downgrading or reverting.

2.2.2.31 event: An occurrence that may require reporting by the utilities defined in this part of ISO/IEC 15068.

The reporting of an event may cause data to be written to stdout, stderr, or to a log file.

2.2.2.32 execution phase: The operations a software administration utility performs that modify the target.

2.2.2.33 exported catalog: Refers to information organized in the exported catalog structure of the standard packaging layout.

It is used for distribution catalogs as well as exporting installed software catalogs using `swlist -c` *catalog*.

Within software definition files of an exported catalog, all data that can be encoded using IRV {1}, shall be. Any such data that cannot be so encoded shall be transformed using UTF-8 {4}.

NOTE: See Section 5.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.2.2.34 exrequisite: The specification in a software object such that it shall not be installed if one or more specific software objects are installed.

2.2.2.35 extended option: The options that can be specified with the `-x` option. These options may be defined in defaults files or options files.

2.2.2.36 filename: A POSIX.1 {2} filename with characters drawn from the POSIX.1 {2} portable filename character set.

NOTE: See 2.2.2.60 of POSIX.1 {2}.

2.2.2.37 filename character string: A sequence of characters from the portable filename character set, not including the `/` (slash) character.

Within software definition files of exported catalogs, all such strings shall be encoded using IRV {1}.

NOTE: See 2.2.2.60 of POSIX.1 {2}.

2.2.2.38 fileset: Defines the files that make up a software object, and is the lowest level of software object that can be specified as input to the software administration utilities.

NOTE: See 3.9.

2.2.2.39 file storage structure: The storage directories in the software packaging layout under which the actual software files for each fileset are located.

NOTE: See Section 5.

2.2.2.40 fully qualified software_spec: A `software_spec` that always identifies a software object unambiguously.

NOTE: See 4.1.4.1.

2.2.2.41 graphical user interface: A means of presenting function to a user through the use of graphics.

All such interfaces are outside the scope of this part of ISO/IEC 15068.

2.2.2.42 hard link: A directory entry, as defined in 2.2.2.17. of POSIX.1 {2}.

2.2.2.43 hexadecimal character string: A sequence of characters from the set of hexadecimal digits, preceded by the two characters `0x` (zero followed by a lowercase "x").

Hexadecimal character strings shall consist only of the following characters:

`0 1 2 3 4 5 6 7 8 9 A B C D E F x`

Within software definition files of exported catalogs, all such strings shall be encoded using IRV {1}.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.2.2.44 host: A machine that contains software managed by this part of ISO/IEC 15068.

NOTE: A host may contain both installed_software and distribution software_collections. The name of the host is the starting point for finding all software on that machine managed by this part of ISO/IEC 15068. The *path* attribute of a software_collection, along with the specification of a host, can be used on the command line to identify a particular software_collection to be managed by this part of ISO/IEC 15068.

2.2.2.45 host character string: A sequence of characters describing a host, as defined in 2.2.2.44.

Within software definition files of an exported catalog, all data that can be encoded using IRV {1} shall be. Any such data that cannot be so encoded shall be transformed using UTF-8 {4}.

2.2.2.46 INDEX file: The file within an exported catalog containing the metadata describing the software objects and attributes for all bundles, products, subproducts and filesets.

NOTE: The format of this file is defined in 5.2.

2.2.2.47 INFO file: For each product and fileset, the file within an exported catalog containing the metadata describing the software_file objects and attributes.

NOTE: The format of this file is defined in 5.2.

2.2.2.48 inheritance: The way in which the attribute definitions of a common object class are used as a part of the definition of other object classes.

The definition of the new object class includes the definition of the common class plus the additional definitions specific to the new object class.

2.2.2.49 installed software: Any software object created by the use of the `swinstall` utility.

2.2.2.50 installed_software: A software_collection containing installed software.

This software is in a state ready for use, or ready to be shared by client systems. A directory path on a system and an installed_software catalog together identify a unique installed_software object.

2.2.2.51 installed_software catalog: The catalog of metadata for an installed_software software_collection.

Unlike a catalog for a distribution object, the storage and format of an installed_software catalog is undefined within this part of ISO/IEC 15068. The ability to dump and restore all or part of an installed_software catalog into an exported catalog structure is included in this part of ISO/IEC 15068.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.2.2.52 installed_software path: The root directory of an installed_software object; the pathname below which all software for that object shall be installed.

2.2.2.53 integer character string: A decimal character string, an octal character string, or a hexadecimal character string.

NOTE: See 2.2.2.19, 2.2.2.63, and 2.2.2.43.

2.2.2.54 interactive: The behavior of a utility or control_script which requires input from the user during its execution.

2.2.2.55 kernel: The nucleus of the operating system.

NOTE: See B.2.2.2 of POSIX.1 {2}.

2.2.2.56 kernel fileset: A fileset in which one or more of the referenced files forms part of the kernel, and denoted by having the value of its *is_kernel* attribute set to `true`.

2.2.2.57 locatable fileset: A fileset for which permission is granted to install the files in a different location as specified by the user, and denoted by having the value of its *is_locatable* attribute set to `true`.

2.2.2.58 locatable software: Software that contains locatable filesets.

2.2.2.59 manager role: Where each task is initiated.

The *manager role* is concerned with taking appropriate action at the completion or failure of a task.

2.2.2.60 metadata: The information kept about software.

It consists of the values of the various attributes of each of the objects.

2.2.2.61 newline string: A white space string consisting only of the `<newline>` character.

NOTE: The term *white space string* is defined in 2.2.2.110 and `<newline>` character is defined in 2.2.2.107 of POSIX.2 {3}.

2.2.2.62 object: An instance in the software hierarchy that can be operated on using the software administration utilities.

2.2.2.63 octal character string: A sequence of characters from the set of octal digits the first of which shall be the digit zero.

Octal character strings shall consist only of the following characters:

```
0 1 2 3 4 5 6 7
```

Within software definition files of exported catalogs, all such strings shall be encoded using IRV {1}.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.2.2.64 options file: A file that can be specified with the `-x` option. This file contains extended option definitions that override default definitions.

NOTE: See 2.2.2.21.

2.2.2.65 packager role: Where software that has been developed is organized in a form suitable for distribution.

2.2.2.66 pathname: A POSIX.1 {2} pathname with characters drawn from the POSIX.1 {2} portable character set.

NOTE: See 2.2.2.60 of POSIX.1 {2}.

2.2.2.67 pathname character string: A sequence of characters from the portable filename character, including the `/` (slash) character.

NOTE: See 2.2.2.60 of POSIX.1 {2}.

Within software definition files of exported catalogs, all such strings shall be encoded using IRV {1}.

2.2.2.68 portable character string: A sequence of characters from the portable character set as defined in 2.2.2.130 and 2.4 of POSIX.2 {3}.

Within software definition files of exported catalogs, all such strings shall be encoded using IRV {1}.

2.2.2.69 prerequisite: The specification in a software object that implies it shall not be installed until after some other software object is installed, and configured until after the other software object is configured.

NOTE: The manner of honoring such a prerequisite is described in 4.5.7.1 and 4.5.7.2.

2.2.2.70 product: A software object used to define a set of related software. Filesets are contained within products.

2.2.2.71 product specification file (PSF): The input file used to define the structure and attributes of software objects and related files to be packaged by the `swpackage` utility.

2.2.2.72 proxy install: A proxy install uses an alternate root directory as the target path.

2.2.2.73 recovery: The ability of the `swinstall` utility, for a failed software install, to return the system to the state that it was in before the failure, including restoring the files.

2.2.2.74 reboot fileset: A fileset which, if installed, requires reboot of the operating system to complete its installation, and denoted by having the value of its *is_reboot* attribute set to `true`.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.2.2.75 rebooting: An implementation-defined procedure generally used to terminate and then restart operations on the target system.

2.2.2.76 role: The context in which an operation is executed.

The utilities in this part of ISO/IEC 15068 require the ability to perform operations on more than one system, perhaps by more than one person. These operations are separated into distinct roles including developer, packager, manager, source, target, and client.

2.2.2.77 selection phase: The set of steps performed by software administration utility to process selections and options.

2.2.2.78 serial medium: A medium that contains a POSIX.1 {2} extended `tar` or extended `cpio` archive.

NOTE: See 10.1.1 and 10.1.2 of POSIX.1 {2}.

2.2.2.79 session: An execution of a software administration command from initiation to completion on all applicable roles.

2.2.2.80 shell token string: A sequence of shell tokens.

A shell token string shall be a portable character string.

NOTE: See 2.2.2.68. Shell tokens are defined in 3.3 of POSIX.2 {3}.

2.2.2.81 software: A generic term referring to software objects or a structured set of files.

This term can refer to the objects forming the hierarchical structure (software objects), or to the actual files and control_files (software files).

NOTE: See 2.2.2.90.

2.2.2.82 software_collection: A grouping of software objects that are managed by the software administration utilities.

Software_collections are the sources and targets of these utilities. This part of ISO/IEC 15068 defines two types of software_collections: installed_software and distributions.

2.2.2.83 software common class: The common class describing the common attributes associated with the hierarchical structure of software objects defined by this part of ISO/IEC 15068.

2.2.2.84 software definition files: The files containing the software structure and detailed attributes for distributions, installed_software, bundles, products, subproducts, filesets, files, and control_files.

This includes the `INDEX` and `INFO` files and the PSF.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

To communicate metadata information relating to both distributions and installed software, software definition files serve as input to, or output from, the various software administration utilities. The format used by software administration utilities to store metadata relating to installed software is undefined.

NOTE: See 5.2.

2.2.2.85 software file: A generic term referring to the files and control_files that are contained within software objects and managed by the utilities in this part of ISO/IEC 15068.

2.2.2.86 software_file common class: The common class that relates the two types of files defined by this part of ISO/IEC 15068, namely the actual files that make up the software, plus the control_files that are executed by the utilities when operating on software.

2.2.2.87 software_files: A generic term referring to file and control_file objects (those that share the same software_file common class).

2.2.2.88 software hierarchy: Hierarchical organization of objects that are managed by the software administration utilities.

2.2.2.89 software location: The directory relative to the installed_software root directory where the relocatable files of the software have been located.

2.2.2.90 software object: An object that inherits attributes of the software common class, meaning a bundle, product, subproduct, or fileset object.

2.2.2.91 software packaging layout: The format for software in a distribution. It contains the metadata for the distribution catalog in a well-defined exported form, as well as the files for the software objects in that distribution.

NOTE: For a detailed description, see Section 5.

2.2.2.92 software pattern match string: A sequence of one or more strings, each made up of a sequence of one or more characters from the shell "Pattern Matching Notation" strings described in 3.13, of POSIX.2 {3}, and with the meaning defined in that clause. If there are two or more strings, the strings are separated by the **|** character.

The match is true if any of the sequences of strings match according to 3.13 of POSIX.2 {3}.

A software pattern match string shall be portable character string.

2.2.2.93 software_spec: A string that is used to identify one or more software objects for input to a software administration utility.

NOTE: See 4.1.4.1.1.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.2.2.94 source: The specification of a source distribution object for a software administration utility. The source host provides a means to locate the source role and the source path is a path accessible to the source host.

2.2.2.95 source host: The host portion of a source specification.

2.2.2.96 source path: The pathname portion of a source specification.

2.2.2.97 source role: Where the software exists in a form suitable for distribution, forming a context for the establishment of a repository of software from which the manager may choose to distribute to targets.

Software exists in the source until it is removed by a task initiated by the manager. The source role provides a repository where software may be stored and provides access for those roles that require the software.

2.2.2.98 subproduct: A software object that is a grouping of software filesets and other subproducts within a product.

NOTE: See 3.10.

2.2.2.99 symbolic link: A type of file that contains a pathname.

Rather than containing data itself, this type of file will resolve to another, as defined by the contained pathname. The way in which this type of file is handled by implementations of this part of ISO/IEC 15068 is undefined.

NOTE: It is not the intention of this part of ISO/IEC 15068 to define symbolic links in a manner inconsistent with POSIX.1 {2}. However, no approved POSIX standard currently contains symbolic links. This definition is a placeholder until such time as an approved standard provides the definition. See POSIX.1a {B21}.

2.2.2.100 system: An implementation of this part of ISO/IEC 15068.

2.2.2.101 target: The specification of a target distribution object, or installed software object, for a software administration utility. The target host provides a means to locate the target role and the target path is a path accessible to the target host.

2.2.2.102 target host: The host portion of a target specification.

2.2.2.103 target path: The pathname portion of a target specification.

2.2.2.104 target role: Where software is installed, removed, listed, and otherwise operated on by the utilities.

For example, when installing software, the target is where software is installed after having been delivered from a source. As another example, the target for a copy operation command refers to the distribution to which products are added. For management operations like removing software, the target refers to either the

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

installed_software objects or the distributions from which software is being removed.

2.2.2.105 update: Installing a newer revision of software than one that is currently installed, into the same location.

This is also referred to as upgrading.

2.2.2.106 vendor: A supplier of packaged software.

This term applies to anyone who creates packaged software, including commercial and non-commercial suppliers, system administrators, and end users.

2.2.2.107 vendor-defined: An item, such as a nonstandard attribute, that is defined by the vendor that created (packaged) the software.

2.2.2.108 vendor-supplied: An item, such as a control file, that is supplied by the creator (packager) of the software.

2.2.2.109 version: A unique identification of software based on the attributes of the software. Version differentiates software objects with the same value of the *tag* attribute.

Versions of bundles or products have the same value of the *tag* attribute and will differ by the value of at least one of *revision*, *architecture*, *vendor_tag*, *location*, or *qualifier* attributes. The *location* and *qualifier* attributes only apply to software in installed_software software_collections.

A fileset is considered a version of another fileset if they have the same *fileset.tag* and their respective products have the same *product.tag*.

2.2.2.110 white space string: A sequence of one or more white space characters (as defined in 2.2.2.191 of POSIX.2 {3}) including `<space>`, `<tab>`, and `<new-line>`.

Within software definition files of exported catalogs, all such strings shall be encoded using IRV {1}.

2.2.2.111 wildcard character: One of `*?[` (asterisk, question mark, open bracket).

Such characters are used in software pattern match strings 2.2.2.92.

2.2.3 Abbreviations

For the purposes of this part of ISO/IEC 15068, the following abbreviations apply:

2.2.3.1 API: Application Programming Interface

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.2.3.2 CLI: Command Line Interface, as defined in 2.2.2.11.

2.2.3.3 C Standard: ISO/IEC 9899: 1990, *Information technology—Programming languages—C* {B13}.

2.2.3.4 CRC: Cyclic Redundancy Check.

2.2.3.5 GUI: Graphical User Interface, as defined in 2.2.2.41.

2.2.3.6 IRV: ISO/IEC 646: 1991, *Information technology—ISO 7-bit coded character set for information interchange,* International Reference Version {1}.

2.2.3.7 newline: A newline string, as defined in 2.2.2.61.

2.2.3.8 OS: Operating System.

2.2.3.9 POSIX.1: ISO/IEC 9945-1: 1990 (ANSI/IEEE Std 1003.1-1990): *Information technology—Portable Operating System Interface (POSIX®)—Part 1: System Application Program Interface (API) [C Language]* {2}.

2.2.3.10 POSIX.1a: IEEE P1003.1a, *Draft Standard for Information technology—Portable Operating System Interface (POSIX®)—Part 1: System Application Program Interface (API) [C Language]* {B21}.

2.2.3.11 POSIX.2: ISO/IEC 9945-2: 1993 (ANSI/IEEE Std 1003.2-1992): *Information technology—Portable Operating System Interface (POSIX®)—Part 2: Shell and Utilities* {3}.

2.2.3.12 POSIX.7.2: This part of ISO/IEC 15068.

2.2.3.13 PSF: product specification file as defined in 2.2.2.71.

2.2.3.14 symlink: symbolic link, as defined in 2.2.2.99.

2.2.3.15 UTF-8: UCS Transformation Format 8, as defined in ISO/IEC 10646-1: 1993, *Information technology—Universal Multiple-Octet Coded Character Set (UCS)—Part 1: Architecture and Basic Multilingual Plane, Amendment 2: UCS Transformation Format 8 (UTF-8), 1996* {4}.

2.2.3.16 white space: A white space string, as defined in 2.2.2.110.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

2.3 Dependencies on Other Standards

2.3.1 Features Inherited From POSIX.1

This subclause describes some of the features provided by POSIX.1 {2} that are assumed to be globally available to all conforming implementations. This subclause does not attempt to detail all the POSIX.1 {2} features that are required by all the utilities defined in this part of ISO/IEC 15068; the utility descriptions point out additional functionality required to provide the corresponding features needed.

The following subclauses describe frequently used concepts. Utility description statements override these defaults when appropriate.

2.3.1.1 File System

The hierarchical directory structure of POSIX.1 {2} is assumed to be available, as well as support for case-sensitive file names. In addition, various file attributes are also assumed to be present, including the following — type, owner, group, mode, uid, gid, mtime, major, and minor.

2.3.1.2 Environment Variables

The existence of environment variables in general is assumed, as well as **PATH**, **LANG**, **LC_ALL**, **LC_CTYPE**, **LC_MESSAGES**, **LC_TIME**, and **TZ**, in particular.

2.3.1.3 Data Interchange Format

The ability to read and write the data interchange formats of POSIX.1 {2} is assumed, including both extended `tar` and extended `cpio`. See 10.1.1 and 10.1.2 of POSIX.1 {2}. See also 5.3.

2.3.2 Features Inherited From POSIX.2

This subclause describes some of the features provided by POSIX.2 {3} that are assumed to be globally available to all systems conforming to this part of ISO/IEC 15068. This subclause does not attempt to detail all of the POSIX.2 {3} features that are required by all the utilities and control scripts defined in this part of ISO/IEC 15068; additional functionality required may be found in the utility descriptions and in 4.1.6.1.

All of the utilities defined in POSIX.2 {3} are required, including the shell interpreter (`sh`). This assures a portable environment for executable control files.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

Documents
Document Sharing Hub

Full document details: ISO IEC 15068-2-1999

Section 3: Software Structures

This section describes the software classes and attributes applicable to software administration. Each utility in Section 4 describes the operations on the software objects including how the values of the attributes affect the behavior of the operations. Whether these operations and behaviors are implemented as procedures on software structures or by other means is undefined within this part of ISO/IEC 15068.

The software administration classes form a hierarchy that consists of distributions, media, installed_software, vendors, bundles, products, subproducts, filesets, control_files, and files.

At each level, this hierarchy is defined by containment attributes that reference objects at lower levels. Operations on objects of lower levels, such as files, are actually enacted by operations on objects of higher levels. For example, files may be created in a distribution by copying a software product.

A "common class" is used to define attributes that are common between related objects. Objects inherit attribute definitions from common classes as well as their individual attributes. This provides a logical relationship between the objects that share the same common class. The software administration common classes are software_collection, software, and software_file.

Objects that share the same common class are also referred to generically as software_collections, software objects, and software_files.

In tables in this section, attributes are listed with various properties. The attributes and their values manifest themselves as part of the utilities defined in Section 4 and the software packaging layout in Section 5.

The names of attributes are as provided. If the underlying host allows for the distinction of case, the attribute names shall be sensitive to case. Where values of attributes are shown, if the underlying host allows for the distinction of case, the values of attributes shall be sensitive to case. If the underlying host does not allow for the distinction of case for either the name or value of an attribute, the way in which case differences are handled is implementation defined.

The attribute tables in this section list the following information:

Attribute
: The name of the attribute, also used as the keyword for the attribute.

Length
: The maximum permitted length of the value of the attribute.

All attribute values in this part of ISO/IEC 15068 are represented only as strings. The length is the maximum permitted length of the value in bytes or, for attributes whose values are lists, the maximum permitted number of items permitted in the list. Since the means of

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

storing such data for installed software is undefined within this part of ISO/IEC 15068, an implementation may store such values internally in different structures for installed software. See 2.2.2.28, 2.2.2.33, and 2.2.2.51.

Permitted Values
The character sequences permitted as values for this attribute.

Default Value
The value of the attribute if the attribute is not specified.

A default value of `None` means the system shall not supply a value in cases where the attribute has not been specified and the attribute is not one whose values are generated dynamically. See 5.1.1 and 5.2.

The attribute tables are broken into the following three groups:

(1) The top group contains the attributes that are used to identify a particular instance.

(2) The middle group contains the rest of the attributes that describe other information or behaviors for the object.

(3) The bottom group contains the attributes that describe the objects contained within this object. The way in which these lists are represented in software definition files is described in 5.2. The way in which these lists are represented by `swlist` is described in 4.6.3.

Beyond this convention, the order of attributes shown in this section is not significant. For any attribute ordering rules, see 5.2. Some attributes do not apply to software objects in both distributions and installed_software objects. See 5.2 for details.

Management of lists of software_collections contained within a host is undefined within this part of ISO/IEC 15068. See 4.1.4.2 for the way in which software_collections are identified relative to a software host.

3.1 Software_Collection

A software_collection is the common class from which distribution and installed_software objects inherit.

A software_collection can contain product and bundle software objects. A software_collection can contain multiple versions of the same product or bundle software objects, namely products or bundles that share the same value for the *tag* attribute.

Each software_collection has a catalog associated with it that contains the metadata describing all software objects in that collection.[4)]

NOTE: For distribution software_collections, the catalog information is stored in the software packaging layout in an exported catalog structure. For installed_software objects, how the catalog information is stored (whether in a file or database, for example) is undefined within this part of ISO/IEC 15068.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

Table 3-1 – Attributes of the Software_Collection Common Class

Attribute	Length	Permitted Values	Default Value
path	Undefined	Pathname character string	Implementation defined
dfiles	64	Filename character string	`dfiles`
layout_version	64	`1.0`	`1.0`
pfiles	64	Filename character string	`pfiles`
bundles	Undefined	List of `bundle_software_spec`s	Empty list
products	Undefined	List of `product_software_spec`s	Empty list

3.1.1 Software_Collection Attributes

The attributes listed in Table 3-1 and described in the following, characterize each instance of the software_collection class, and are inherited by each instance of the distribution and installed_software classes:

bundles A list of `bundle_software_spec`s.

Each `software_spec` shall refer to a bundle. Each `software_spec` shall be fully qualified. See 4.1.4.1 for the syntax of `software_spec`.

dfiles The name of the directory in the exported catalog structure below which any attributes stored as files for the software_collection are stored (see 5.1).

layout_version
This attribute, and its value, are included for future use.

path The identifier for a particular software collection on a host.

The value of the path attribute shall be an absolute path. The default value of this attribute is implementation defined. See 4.1.5.2.

pfiles The name of the directory in the exported catalog structure below which any control_files, and attributes stored as files, for the product are stored (see 5.1).

products A list of `product_software_spec`s.

Each `software_spec` shall refer to a product. Each `software_spec` shall be fully qualified. See 4.1.4.1 for the syntax of `software_spec`.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

3.2 Distribution

A distribution contains product and bundle software objects. It is contained on a distribution media or may be part of the file store of a system. The distribution may contain a variety of software products and bundles, and that software may be applicable to a variety of hardware architectures or operating systems.

The distribution class inherits attributes from the software_collection common class.

A particular distribution object is identified within a host by the *path* attribute. For distributions, the *path* attribute is the pathname to the directory containing a distribution in the directory format of the software packaging layout, or a file or device file containing a distribution in a serial format of the software packaging layout.

Distributions can contain more than one version of a product or bundle. A version is uniquely identified within a distribution by the values of the *revision*, *vendor_tag*, and *architecture* attributes.

Table 3-2 – Attributes of the Distribution Class

Attribute	Length	Permitted Values	Default Value
uuid	64	Portable character string	Empty string
media	Undefined	List of media *sequence_number* values	Empty list

3.2.1 Distribution Attributes

The attributes listed in Table 3-2 and described in the following, along with the attributes listed in Table 3-1, characterize each instance of the distribution class:

media A list of *media.sequence_number* values for the distribution if the distribution spans multiple media. Each medium in a distribution shall have its *media.sequence_number* in the **INDEX** file defined for that medium. See 5.3. An implementation may include definitions for all media in the global **INDEX** file found on the first medium in the distribution. The *media.sequence_number* for the first medium in the distribution shall be **1** and shall be the first item in the list.

uuid A string that should uniquely identify a distribution.

The way in which a unique string is generated is undefined. This attribute is used for determining whether subsequent media are from the same set as the one that an install or copy started with. This attribute shall be defined for distributions that span multiple media.

Documents Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

3.3 Media

The media class is used to describe the media attributes for distributions that span multiple media.

Table 3-3 – Attributes of the Media Class

Attribute	Length	Permitted Values	Default Value
sequence_number	64	Portable character string	`1`

3.3.1 Media Attributes

The attribute listed in Table 3-3 and described in the following, characterizes each instance of the media class:

sequence_number

Identifies a particular media when a distribution spans multiple media.

It is used for identifying the correct medium on which to find the distribution files when the distribution spans multiple media.

3.4 Installed_Software

The installed_software class is used to describe the bundle and product software that has been installed on a file system.

The installed_software class inherits attributes from the software_collection common class.

A particular installed_software object is identified within a host by both the *path* attribute (defined in the software_collection class) and the *catalog* attribute. For installed_software objects, the *path* attribute is the root directory for the installed_software object below which all the software files are installed.

An installed_software object can contain multiple versions of a product or bundle. Multiple product and bundle versions are distinguished by the same attributes as distribution products, plus the user-specifiable *location* and *qualifier* attributes. Multiple product versions may be installed at the same time in an installed_software object. Different product versions may be installed into different locations, and different filesets from different product versions may be installed in the same location.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

Table 3-4 – Attributes of the Installed Software Class

Attribute	Length	Permitted Values	Default Value
catalog	Undefined	Portable character string	Undefined

3.4.1 Installed_Software Attributes

The attribute listed in Table 3-4 and described in the following, along with the attributes listed in Table 3-1, characterizes each instance of the installed_software class:

catalog Along with the *path* attribute, identifies a single installed_software object.

Different installed_software objects may have the same value for the *path* attribute if and only if the value of their *catalog* attributes are different.

The *catalog* attribute is evaluated relative to the *path* attribute. It may be a POSIX.1 {2} pathname or other identifier: together they form the key to the undefined catalog storage for this installed_software object.

3.5 Vendor

The vendor class is used to describe the attributes of the vendors associated with products and bundles.

Each product or bundle identifies a vendor with a vendor_tag that identifies a particular vendor object. The *vendor_tag* attribute is used to distinguish products and bundles from different vendors that share the same product or bundle *tag*.

Table 3-5 – Attributes of the Vendor Class

Attribute	Length	Permitted Values	Default Value
tag	64	Filename character string	Empty string
title *description*	256 Undefined	Portable character string Portable character string	Empty string Empty string

Documents Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

3.5.1 Vendor Attributes

The attribute listed in Table 3-5 and described in the following, characterizes each instance of the installed_software class:

description A more detailed description of the vendor or information about the vendor.

tag A short identifying name of the vendor that supplied the product.

This attribute is used to to distinguish products and bundles from different vendors, and for resolving software specifications. Each software vendor should attempt to have a unique value for the *tag* attribute.

title A longer name of the vendor that supplied the product. It is used for presentation purposes.

3.6 Software

Software is the common class from which products, bundles, filesets and subproducts inherit.

Table 3-6 – Attributes of the Software Common Class

Attribute	Length	Permitted Values	Default Value
tag	64	Filename character string	None
create_time	16	Integer character string	None
description	Undefined	Portable character string	Empty string
mod_time	16	Integer character string	None
size	32	Integer character string	None
title	256	Portable character string	Empty string

3.6.1 Software Common Attributes

The attributes listed in Table 3-6 and described in the following, characterize each instance of the software common class, and are inherited by each instance of the product, bundle, fileset, and subproduct classes:

create_time A value that shall be set by the implementation to be the time that the catalog information for this object was first written.

Time shall be represented as seconds since the Epoch, as defined in 5.6.1.3 of POSIX.1 {2}.

description A more detailed description of the software object.

mod_time A value that shall be set by the implementation to be the time that the catalog information for this object was last written.

Documents
Document Sharing Hub
Full document details: ISO IEC 15068-2-1999

Time shall be represented as seconds since the Epoch, as defined in 5.6.1.3 of POSIX.1 {2}.

size The sum of the sizes in bytes of all files and control_files contained within the software object.

For objects other than filesets, the value is computed dynamically as required. See 5.2.6, 5.2.7, and 5.2.8.

tag A short name associated with the software object.

It is the one attribute that is always required to identify a software object. For more information on software selections, see 4.1.4.1.

title A longer name associated with the software object, used for display purposes.

3.7 Products

Products can contain filesets, which can be grouped into subproducts. Products are named by their *tag* attributes. A particular product object is uniquely identified within a software_collection by the *tag* attribute and by the version distinguishing attributes. The attributes that uniquely distinguish a particular product version within a software_collection are *revision*, *architecture*, *vendor_tag*, *location*, and *qualifier*.

The product class shall inherit the attributes of the software common class.

See 4.1.4.1.2 on software compatibility.

3.7.1 Product Attributes

The product attributes listed in Table 3-7 and described in the following, along with the attributes listed in Table 3-6, characterize each instance of the product class:

all_filesets This is a list of all filesets defined for the product, as opposed to what is currently installed, described by the *filesets* attribute. The *all_filesets* attribute is used to determine completeness of this product when another software object has a dependency on this product. In checking a product prerequisite or corequisite, the existence of a *fileset.tag* in *all_filesets* that is not actually **`installed`** or **`available`** indicates that the dependency is not satisfied.

This does not affect exrequisites as they test whether any of the contents of the dependency specification are present instead of all of the contents tested for prerequisites or corequisites.

architecture A vendor-defined string used to distinguish variations of a product.

It is used for presentation purposes and for resolving software specifications. If a product with the same value of the *revision*

Documents Document Sharing Hub
Full document details: ISO IEC 15068-2-1999