BRITISH STANDARD

BS ISO/IEC 19770-2:2009

Information technology - Software asset management

Part 2: Software identification tag

ICS 35.080

NO COPYING WITHOUT BSI PERMISSION EXCEPT AS PERMITTED BY COPYRIGHT LAW

Document Sharing Hub
British Standards

Full document details: BS ISO/IEC 19770-2:2009

National foreword

This British Standard is the UK implementation of ISO/IEC 19770-2:2009.

The UK participation in its preparation was entrusted to Technical Committee IST/15, Software and systems engineering.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Compliance with a British Standard cannot confer immunity from legal obligations.

This British Standard was published under the authority of the Standards Policy and Strategy Committee on 28 February 2010

© BSI 2010

ISBN 978 0 580 54959 5

Amendments/corrigenda issued since publication

Date	Comments

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 19770-2:2009

INTERNATIONAL STANDARD

ISO/IEC 19770-2

First edition
2009-11-15

Information technology — Software asset management —

Part 2:
Software identification tag

Technologies de l'information — Gestion de biens de logiciel —

Partie 2: Étiquette d'identification du logiciel

Reference number
ISO/IEC 19770-2:2009(E)

Documents
Document Sharing Hub
© ISO/IEC 2009

Full document details: BS ISO/IEC 19770-2:2009

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

COPYRIGHT PROTECTED DOCUMENT

© ISO/IEC 2009

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 19770-2:2009

Table of Contents

Page

Foreword v

Introduction vi

1 Scope 1
1.1 Purpose 1
1.2 Field of application 1
1.3 Limitations 1

2 Conformance 2
2.1 General 2
2.2 Product conformance 2
2.3 Organizational conformance 5
2.4 Agreement compliance 6

3 Normative references 6

4 Terms, definitions and abbreviated terms 6
4.1 Terms and definitions 6
4.2 Abbreviated terms 12

5 Alignment and rationalization with prior standards 12
5.1 Statement of alignment for this part of ISO/IEC 19770 12
5.2 Alignment with ISO/IEC 19770-1:2006 Information technology — Software asset management — Part 1: Processes 12
5.3 Alignment with ISO/IEC 20000-1:2005 Information technology – Service management – Part 1: Specification 13
5.4 Alignment with ISO/IEC 20000-2:2005 Information technology — Service management — Part 2: Code of practice 14

6 Implementation of software identification tagging processes 14
6.1 General requirements and guidance 14
6.2 Software identification tagging life cycle: operational breakdown 22

7 Platform requirements and guidance 24
7.1 Types of platforms 24
7.2 Basic platform services 25
7.3 Virtual environments 25
7.4 Virtual machines 26
7.5 Support for software installed on removable media 26
7.6 Hardware and platform identification 26

8 Elements 27
8.1 General 27
8.2 Element names 27
8.3 Mandatory elements 28
8.4 Optional elements 33
8.5 Extended elements 60
8.6 Data type definitions 61

Annex A (informative) **Software identification tagging principles** 67

Annex B (informative) **Software provider use cases and guidance** 73

Annex C (informative) **Tool provider use cases and guidance** 78

Annex D (informative) **Software consumer use cases and guidance** 81

Annex E (informative) **Software identification tags for items other than software** 85

Annex F (informative) **Copyright and software identification tags** 86

Documents
Document Sharing Hub

© ISO/IEC 2009 – All rights reserved

Full document details: BS ISO/IEC 19770-2:2009

Annex G (normative) **XML schema definition (XSD)** **87**
Annex H (informative) **Extended examples** **95**

Figures

Page

Figure 1 — Software identification tag lifecycle **22**

Figure A.1 — Life cycle of a software identification tag **67**

Tables

Page

Table 1 - Examples of regid values **15**

Table 2 - Examples of tag locations on different platforms **16**

Table 3 - Microsoft Vista® APIs for software identification tag management **16**

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 19770-2:2009

Foreword

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. 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.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 19770-2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 7, *Software and systems engineering*.

ISO/IEC 19770 consists of the following parts, under the general title *Information technology — Software asset management*:

— *Part 1: Processes*

— *Part 2: Software identification tag*

— *Part 3: Software entitlement tag*

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

Introduction

This part of ISO/IEC 19770 provides an International Standard for software identification tags. The software identification tag is an XML file containing authoritative identification and management information about a software product. The software identification tag is installed and managed on a computing device together with the software product. The tag may be created as part of the installation process, or added later for software already installed without tags. However, it is expected more commonly that the tag will be created when the software product is originally developed, and then be distributed and installed together with the software product. Having the tag available from the beginning allows for the more effective management of distribution and repackaging external to the software consumer, and then of release management within the software consumers organization.

This part of ISO/IEC 19770 supports software asset management processes as defined in ISO/IEC 19770-1. It is also designed to work together with the future ISO/IEC 19770-3 which will provide an International Standard for software entitlement tags.

Software identification tags will benefit all stakeholders involved in the creation, licensing, distribution, releasing, installation, and on-going management of software. Key benefits associated with software identification tags include:

a) The ability to consistently and authoritatively identify software products that need to be managed for any purpose, such as for licensing, upgrading, packaging or for the specification of dependencies. Software identification tags provide the meta-data necessary to support more accurate identification which differentiates this approach from traditional file-oriented identification techniques.

b) The ability to identify groups or suites of software products in the same way as for individual software products, enabling entire groups or suites of software products to be managed with the same flexibility as for individual products.

c) Facilitation of de facto standardization between different software creators, and within software creator organizations, of how different versions of software are identified, allowing for better identification and management by software consumers of those different versions; for example, being able to distinguish between free-standing versions and versions which are components of suites, upgrade paths, etc.

d) Facilitation of automated approaches to license compliance, using information both from the software identification tag and from the software entitlement tag as will be specified in ISO/IEC 19770-3.

e) The ability to provide comprehensive information about the structural footprint of packages, i.e. the list of components such as files and system settings associated with that package, in order to link package-level management with file-level management.

f) The ability to provide information about how to identify if a particular software package is being actively used or not.

g) The ability to deal with the complexities of software installed on removable or shared storage, or in virtual environments (subject to the evolving ability of platforms and installers to identify devices and environments).

h) The ability to reflect within the software identification tag the identities and requirements of different entities, including software creators, software licensors, packagers, distributors external to the software consumer, release managers within the software consumer, and those responsible for installing and managing software on an on-going basis.

i) The ability to allow for the validation of any of this information through the optional use of digital signatures by anyone creating or modifying information in the software identification tag.

Documents
Document Sharing Hub

© ISO/IEC 2009 – All rights reserved

Full document details: BS ISO/IEC 19770-2:2009

j) The ability for entities besides the software creators (e.g. independent providers, or in-house personnel) to create software identification tags for legacy software, and also for software from software creators who do not provide software identification tags themselves.

k) The ability of this International Standard to evolve in informal and formal ways, as common approaches become accepted throughout industry for dealing with additional types of information not currently covered by this part of ISO/IEC 19770, such as for product activation.

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 19770-2:2009

Information technology — Software asset management —

Part 2:
Software identification tag

1 Scope

1.1 Purpose

This part of ISO/IEC 19770 establishes specifications for tagging software to optimize its identification and management.

1.2 Field of application

This part of ISO/IEC 19770 applies to:

a) Platform providers: These are the entities which are responsible for the computer or hardware device and/or associated operating system, or virtual environment, on which software may be installed or run. Platform providers which support this part of ISO/IEC 19770 additionally provide tag management capabilities at the level of the platform or operating system.

b) Software providers: These are the entities that create (“software creators”), package (“software packagers”) or license (“software licensors”) software for distribution or installation. These include software manufacturers, independent software developers, consultants, and repackagers of previously manufactured software. They may also be in-house software developers.

c) Tag providers: These are the entities that create (“tag creators”) or modify (“tag modifiers”) software identification tags. A tag provider may be part of the software provider organization, or may be a 3rd party organization or the software consumer.

d) Tag tool providers: These are the entities that may provide any number of tools that create, modify or use software identification tags. These tools include development environments that provide automatically generated software identification tags, installation tools that may create and/or modify tags on behalf of the installation process as well as desktop management tools that may create tags for software that does not have a tag and/or modify tags with release details throughout the software lifecycle. See Annex C for details on how tool providers are likely to use software identification tags.

e) Software consumers: These are the entities that purchase, install and/or otherwise consume software, and who are intended as one of the major beneficiaries of the improved information provided by the software identification tag as specified in this part of ISO/IEC 19770. See Annex D for details on how software consumers are likely to use software identification tags.

1.3 Limitations

This part of ISO/IEC 19770 does not detail SAM processes required for reconciliation of software entitlements with software identification tags.

This part of ISO/IEC 19770 does not specify product activation or launch controls.

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 19770-2:2009

This part of ISO/IEC 19770 is not intended to conflict either with any organization's policies, procedures or standards or with any national laws and regulations. Any such conflict should be resolved before using this part of ISO/IEC 19770.

2 Conformance

2.1 General

Conformance can apply to a product or an organization. For organizational conformance, the scope defined shall cover both the organizational scope as well as the products that are included in the scope.

If a claim of conformance is made for a product or organization, the claim shall specify the scope for which the conformance was tested.

Conformance throughout this clause is most often defined in terms of complying with the requirements of 6.1, 8.3, 8.4, and 8.5. Requirements for platform conformance are also specified in 7.2. There are also normative requirements specified in other subclauses of Clauses 6 and 7, indicated by the use of the word “shall”, but these are not included in the coverage of statements of conformance, except to the extent that they are also included in 6.1, 7.2, 8.3, 8.4, or 8.5. Statements including the word ‘should’ are recommendations but not mandatory.

2.2 Product conformance

2.2.1 Example reasons for product conformance

There are a number of reasons for an organization to seek individual product conformance to this part of ISO/IEC 19770. This may be sought when a specific product is being provided for a market that requires conformance (for example, if government organizations require products to conform to this part of ISO/IEC 19770 in order to be included on a project). It might also be desired by platform providers who want to provide a more secure and auditable tag storage that can be used to identify definitively which end-users installed which software packages.

2.2.2 Product scope

There shall be a clear statement for product scope describing, in unambiguous terms, the software products to which it applies and, where appropriate, clarifying the products to which it does not apply. The product conformance scope may be defined in any way considered appropriate, such as for a specific software product, for all software products, for all software products on specific platforms, for the software products of specified manufacturers and/or for all software products created after a specified date, as long as it is unambiguous. In the case of a product which creates or modifies software identification tags, the scope shall be the product itself and all software produced or modified by the product when tag-conformity functionality is enabled.

2.2.3 Software product conformance

Full conformance for a software product is achieved in one of two ways:

a) For a product which is installable, full conformance is achieved by demonstrating that all software identification tags installed by it at installation shall comply with all mandatory requirements of this part of ISO/IEC 19770, as specified in 6.1 and 8.3. If optional or extended tag elements are used these shall also comply with requirements as specified in 8.4 and 8.5.

   This conformance shall be demonstrated by performing equivalence partitioning with the exit criteria that all tests pass and 100 % equivalence partition coverage of the tag creation/installation is achieved. Equivalence partitions shall be derived from the statement of product scope.

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

If the software product consists of a package of other software products, then the software product shall retain all component tags and reference all child tag elements, which, under any circumstances, still need to be identified separately (for the purpose of licensing, security or other).

b) For a product that is distributable but not yet installed, full conformance is achieved by demonstrating that distributable builds are issued with a unique tag that shall comply with all mandatory requirements of this part of ISO/IEC 19770, as specified in 6.1 and 8.3. If optional or extended tag elements are used these shall also comply with requirements as specified in 8.4 and 8.5. The exception to this is that any mandatory elements which are installation-specific are not included.

   This conformance shall be demonstrated by performing equivalence partitioning with the exit criteria that all tests pass and 100 % equivalence partition coverage is achieved. Equivalence partitions shall be derived from the statement of product scope.

   If the software product consists of a package of other software products, then the software product shall retain all component tags and reference all child tag elements which under any circumstances still need to be identified separately (for the purpose of licensing, security or other).

2.2.4 Third party software identification tag conformance

Third party tag provider organizations may undertake the process of creating software identification tags for any software packages that do not include such tags. This may be done for older software products, shareware/freeware type products, or for companies that decide not to follow this part of ISO/IEC 19770. These tags may be provided to organizations to assist in their software discovery and identification procedures.

Full conformance for third party created software identification tags is achieved by demonstrating that all software identification tags produced by the organization comply with all mandatory requirements of this part of ISO/IEC 19770, as specified in 6.1 and 8.3. If optional or extended tag elements are used these shall also comply with requirements as specified in 8.4 and 8.5. Any new data that is added shall conform to the same standards as those required for installable software conformance.

Conformance for third party created software identification tags requires that the tag providers demonstrate that the software_ids they create are unique, and use consistent values for the identification of software providers. The expectation is that the tag providers will maintain a list of unique software providers for all tags created, and that the list includes a consistent software provider regid (that references the provider's domain) and a unique ID (which may be a GUID) for each reference and that these details are used consistently in the created tags.

This conformance shall be demonstrated by performing equivalence partitioning with the exit criteria that all tests pass and 100 % equivalence partition coverage of the tag production is achieved. Equivalence partitions shall be derived both from the range of software that the tag tool shall work on and the corresponding statements of product scope.

2.2.5 Software installer product conformance

Full conformance for a software installer product is achieved by demonstrating that all software identification tags installed by it at installation comply with all mandatory requirements of this part of ISO/IEC 19770, as specified in 6.1 and 8.3. If optional or extended tag elements are used these shall also comply with requirements as specified in 8.4 and 8.5.

This conformance shall be demonstrated by performing equivalence partitioning with the exit criteria that all tests pass and 100 % equivalence partition coverage of the tag creation/installation is achieved. Equivalence partitions shall be derived both from the range of software that is installed and the corresponding statements of product scope.

If the software being installed consists of a package of other software products, then the software product shall retain all component tags and reference all child tag elements which under any circumstances still need to be identified separately (for the purpose of licensing, security or other).

Documents
Document Sharing Hub

© ISO/IEC 2009 – All rights reserved

Full document details: BS ISO/IEC 19770-2:2009

Existing tag values that are provided with distributable software shall not be modified in any way, with some specific exceptions. If a distributed software identification tag is found to be corrupted and that software identification tag does not provide a "validation" routine to fix the tag, a software product may provide options for handling this type of exception that a SAM practitioner can authorize. Based on actions specified by the SAM practitioner, the handling of such exceptions may include actions such as fixing the software identification tag if it is corrupt, deleting the software identification tag if it no longer belongs on the device, or modifying the software identification tag to specify that the software is no longer installed on the device. Should any modifications of the tag be specified by the user, these actions shall be logged and retained by the software product.

It is expected that such products will have the capability to turn this functionality on or off. A statement of product conformance shall apply only to the product with this functionality turned on.

2.2.6 Tag tool conformance

Full conformance for a tag tool is achieved in one of two ways:

a) Full conformance for a tag tool that installs or modifies installed software identification tags independent of software installation is achieved by demonstrating that all software identification tags installed or modified by the product comply with all mandatory requirements of this part of ISO/IEC 19770, as specified in 6.1 and 8.3. If optional or extended tag elements are used these shall also comply with requirements as specified in 8.4 and 8.5. Any new data that is added shall conform to the same standards as those required for installable software conformance.

   This conformance shall be demonstrated by performing equivalence partitioning with the exit criteria that all tests pass and 100 % equivalence partition coverage of the tag production is achieved. Equivalence partitions shall be derived both from the range of software that the tag tool shall work on and the corresponding statements of product scope.

   If the software being installed consists of a package of other software products, then the software product shall retain all component tags and reference all child tag elements which under any circumstances still need to be identified separately (for the purpose of licensing, security or other).

   Existing tag values that are provided with distributable software shall not be modified in any way, with some specific exceptions. If a distributed software identification tag is found to be corrupted and that software identification tag does not provide a "validation" routine to fix the tag, a software product may provide options for handling this type of exception that a SAM practitioner can authorize. Based on actions specified by the SAM practitioner, the handling of such exceptions may include actions such as fixing the software identification tag if it is corrupt, deleting the software identification tag if it no longer belongs on the device, or modifying the software identification tag to specify that the software is no longer installed on the device. Should any modifications of the tag be specified by the user, these actions shall be logged and retained by the software product.

   It is expected that such products will have the capability to turn this functionality on or off. A statement of product conformance shall apply only to the product with this functionality turned on.

b) For a tag tool that discovers, collects, reports on and uses tags (such as discovery tools, desktop management tools or SAM reconciliation tools), full conformance is achieved by demonstrating the following.

   1) That all tags available on a computing device are collected. This includes tags that are stored in the common system location as well as tags that are located in the top level directories of software installations.

   2) That all tags collected from computing devices and stored in the tool's repository can be shown to include exactly the same information as the contents of the tag located on the computing device from which it was originally collected.

   3) If a tag is digitally signed and the corresponding public key is available, that the tool validates the signature and the information that has been signed.

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

This conformance shall be demonstrated by performing equivalence partitioning with the exit criteria that all tests pass and 100 % equivalence partition coverage of the tag collection/validation is achieved. Equivalence partitions shall be derived both from the range of software that the tool shall analyze and the corresponding statements of product scope.

If the software being installed consists of a package of other software products, then the software product shall retain all component tags and reference all child tag elements which under any circumstances still need to be identified separately (for the purpose of licensing, security or other).

Existing tag values that are provided with distributable software shall not be modified in any way, with some specific exceptions. If a distributed software identification tag is found to be corrupted and that software identification tag does not provide a "validation" routine to fix the tag, a software product may provide options for handling this type of exception that a SAM practitioner can authorize. Based on actions specified by the SAM practitioner, the handling of such exceptions may include actions such as fixing the software identification tag if it is corrupt, deleting the software identification tag if it no longer belongs on the device, or modifying the software identification tag to specify that the software is no longer installed on the device. Should any modifications of the tag be specified by the end-user, these actions shall be logged and retained by the software product.

It is expected that such products will have the capability to turn this functionality on or off. A statement of product conformance shall apply only to the product with this functionality turned on.

2.2.7 Platform conformance

Full conformance for a platform's tag functionality is achieved by demonstrating that it can store software identification tag data centrally and provide the following services with integrity, as specified in 7.2.

a) Basic functionality: add, modify, read, and delete tag data.

b) Security: determine which end-user can read, create, delete and modify software identification tags.

c) Audit functionality: identify which end-user installed, modified or removed a given software configuration item and when the modification occurred.

This conformance shall be demonstrated by performing equivalence partitioning with the exit criteria that all tests pass and 100 % equivalence partition coverage of the tag storage is achieved. Equivalence partitions shall be derived both from the range of software that the platform shall host and the corresponding statements of product scope.

2.3 Organizational conformance

2.3.1 Example reasons for organizational conformance

Organizations could want to conform to this part of ISO/IEC 19770 for a number of reasons. For example, software providers could want to promote their software products as being easier to manage. Also, software consumers could want to show that they are actively managing their software assets and that they can provide accurate information to any reconciliation or audit request.

2.3.2 Organizational scope

There shall be a clear statement for the organizational scope describing, in unambiguous terms, the organizational structure to which it applies and, where appropriate, clarifying the areas to which it does not apply. A statement of organizational scope shall be accompanied by a statement of software product scope.

2.3.3 Software provider conformance

Full conformance for a software provider is achieved by the organization demonstrating that all software within the scope meets the relevant product conformance requirements, as specified in 2.2.3.

Documents
Document Sharing Hub

© ISO/IEC 2009 – All rights reserved

Full document details: BS ISO/IEC 19770-2:2009

2.3.4 Tag tool provider conformance

Full conformance for a tag tool provider is achieved by an organization demonstrating that all software within the scope meets the relevant tool conformance requirements, as specified in 2.2.6.

Furthermore, in order to claim tag tool provider conformance, all tag tools produced by the organization shall be included in the product scope.

2.3.5 Software consumer conformance

Full conformance for an organization that installs software is achieved by demonstrating that there are software identification tags in place for all software in the software consumer organization's product scope and that the software identification tags comply with all mandatory requirements of this part of ISO/IEC 19770, as specified in 6.1 and 8.3. If optional or extended tag elements are used, these shall also comply with requirements in 8.4 and 8.5.

2.4 Agreement compliance

This part of ISO/IEC 19770 may be used to help develop an agreement between a software provider and a software consumer, in which case clauses of this part of ISO/IEC 19770 can be selected for incorporation into the agreement, with or without modification. In such an instance, it is necessary for both parties to comply with their agreement rather than conform to this part of ISO/IEC 19770.

NOTE ISO/IEC's copyright and patent policy extends to all of this part of ISO/IEC 19770 and contents thereof. However, for the specific use of agreement compliance, there is no need to obtain copyright permission.

3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

IETF RFC 3986, *Uniform Resource Identifier (URI): Generic Syntax,* 2005

IETF RFC 4646, *Tags for Identifying Languages,* 2006

W3C Recommendation, *XML Signature Syntax and Processing (Second Edition)*, 2008

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

UNSPSC, *The United Nations Standard Products and Services Code*

4 Terms, definitions and abbreviated terms

4.1 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

4.1.1
application
system for collecting, saving, processing, and presenting data by means of a computer

NOTE The term application is generally used when referring to a component of software that can be executed.

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

4.1.2
bundle
grouping of products which is the result of a marketing/licensing strategy to sell entitlements to multiple products as one purchased item

NOTE 1 A bundle can be referred to as a "suite", if the products are closely related and typically integrated (such as an office suite containing a spreadsheet, word processor, presentation and other related items).

NOTE 2 Bundles can also refer to software titles that are less closely related such as a game, a virus scanner and a utility "bundled" together with a new computer, or to groups of entitlements, such as multiple entitlements for a backup software product.

4.1.3
component
entity with discrete structure, such as an assembly or software module, within a system considered at a particular level of analysis

NOTE Component refers to a part of a whole, such as a component of a software product, a component of a software identification tag, etc.

4.1.4
computing device
functional unit that can perform substantial computations, including numerous arithmetic operations and logic operations without human intervention

NOTE A computing device can consist of a stand-alone unit, or several interconnected units. It can also be a device that provides a specific set of functions, such as a phone or a personal organizer, or more general functions such as a laptop or desktop computer.

4.1.5
configuration item
CI
item or aggregation of hardware or software or both that is designed to be managed as a single entity

NOTE Configuration items may vary widely in complexity, size and type, ranging from an entire system including all hardware, software and documentation, to a single module, a minor hardware component or a single software package.

4.1.6
configuration management database
CMDB
database containing all the relevant details of each configuration item and details of the important relationships between them

4.1.7
customer
end-users or organizations for which a software publisher designs and develops software and sells entitlements to use that software

4.1.8
element
component of a software identification tag that provides information related to the software represented by the tag

NOTE The different types of elements are defined in 8.3, 8.4 and 8.5.

4.1.9
end-user
person (or persons) who operate or interact directly with a computing device to manage or use software packages

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

4.1.10
equivalence partitioning
software testing process that identifies representative groups of input values that are processed in the same manner by software products allowing for a sampling of each representative group to validate the outcome resulting in a reduced number of test cases while ensuring full coverage of all test cases

4.1.11
extensible markup language
XML
license-free and platform-independent markup language that carries rules for generating text formats that contain structured data

4.1.12
globally unique identifier
GUID
16-byte string of characters that is generated in a manner that gives a high probability that the string is unique in any context

NOTE 1 Other globally unique identifier algorithms can be used in some situations. In general, alternative algorithms use Uniform Resource Identifier (URI) based structures, so the id owner's registration identifier (regid) is included in the identifier.

NOTE 2 GUID as an all capitalized term refers specifically to the 16 byte version. If the term is in lowercase (guid), it refers to a general algorithm that can use either a URI, or a 16-byte-based identifier.

4.1.13
legacy software
software originally created without software identification tags

4.1.14
line of business application developer
person or company specializing in developing applications providing specific functions for a particular business operation

4.1.15
MD5
Message-Digest algorithm 5
algorithm that is a widely-used cryptographic hash function with a 128 bit hash value often used to identify if two files contain the same data

4.1.16
package
set of related components that are combined into a single distributable item

NOTE For example, a software package would be a set of files that can be used to install software on a computing device and can be distributed via CD or electronic means.

4.1.17
platform
computer or hardware device and/or associated operating system, or a virtual environment, on which software can be installed or run

NOTE Examples of platforms include Linux™, Microsoft Vista®, and Java™.

4.1.18
platform provider
organization responsible for the platform

NOTE The platform provider is typically the vendor of the relevant operating system or virtual environment.

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

4.1.19
product
complete set of computer programs, procedures and associated documentation and data designed for delivery to a software consumer

NOTE The terms "product" and "software package" are used interchangeably depending on the context of the item described.

4.1.20
registration identifier
regid
identifier created from a domain name and the date when the domain was owned by a specific individual or company, allowing an individual or company to have their own unique namespace and be their own registration authority for all software configuration items they publish without requiring a separate industry based registration authority

4.1.21
release
collection of new and/or changed configuration items which are tested and introduced into a production environment together

4.1.22
release manager
individual responsible for the collection of new and/or changed configuration items which are tested and introduced into an organization's live production environment

4.1.23
SAM owner
individual at a senior organization-wide level who is identified as being responsible for SAM

4.1.24
SAM practitioner
individual involved in the practice or role of managing software assets

NOTE A SAM practitioner is often involved in the collection or reconciliation of software inventory and/or software entitlements.

4.1.25
software
all or part of the programs, procedures, rules, and associated documentation of an information processing system

4.1.26 Software Asset Management
SAM
effective use, control and protection of software assets within an organization

4.1.27
software consumer
organization or person who buys entitlements to use a software package

4.1.28
software creator
person or organization that creates a software product or package

NOTE This entity might or might not own the rights to sell or distribute the software.

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

4.1.29
software developer
person who creates software that can perform a specified set of actions

NOTE Often a software developer works with other developers for a software manufacturer to create commercial applications. A software developer can also often work as an in-house developer of software for use by the software developer's own organization.

4.1.30
software entitlement
legal ownership of software license use rights as defined through agreements between a software purchaser and the software copyright holder

NOTE Effective use rights take into account any contracts and all applicable licenses, including full licenses, upgrade licenses and maintenance agreements.

4.1.31
software identification tag
file comprised of mandatory elements, optional elements and extended information containing authoritative identification information about a software configuration item

NOTE For mandatory elements see 8.3, for optional elements see 8.4, for extended information see 8.5.

4.1.32
software license
legal rights to use software in accordance with terms and conditions specified by the software copyright owner

NOTE "Using a software product" can include: accessing, copying, distributing, installing and executing the software product, depending on that product's terms and conditions.

4.1.33
software licensor
person or organization who owns the rights to issue a software license for a specific software package

4.1.34
software manufacturer
group of people or organization that develops software, typically for distribution and use by other people or organizations

4.1.35
software package
complete and documented set of programs supplied for a specific application or function

NOTE In this part of ISO/IEC 19770, the term software package refers to the set of files associated with a specific set of business functionality that can be installed on a computing device and has a set of specific licensing requirements. In this part of ISO/IEC 19770, the terms "product" and "software package" are used synonymously depending on the context of the item described.

4.1.36
software packager
entity that re-packages or bundles software created by others

NOTE This can be done by a value added reseller who bundles a software package to work with an embedded system, or by a software reseller who is licensed to combine a number of different software products into a single bundle.

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

4.1.37
software provider
entity that creates (software creator), modifies (software modifier) or licenses (software licensor) software for distribution or installation

NOTE This includes software manufacturers, independent software developers, consultants and repackagers of previously manufactured software. IT can also represent in-house software developers.

4.1.38
software publisher
person, group or company that packages and distributes software and might or might not be the software manufacturer

4.1.39
tag creator
entity that initially creates the software identification tag

NOTE This entity can be part of the organization that created the software, in which case the tag creator and software creator will be the same. The tag creator can also be a third party organization unrelated to the software creator (such as in the case where tags are created for legacy software).

4.1.40
tag modifier
software packager or software consumer that modifies a tag after it has been created

NOTE Modification of any tag is limited to the elements that the software licensor has authorized and is done based on license or contractual agreements with the tag creator and/or software creator. The tag modifier can be allowed to add values to a software identification tag (such as the case of a reseller adding details about where the product was purchased), or can be allowed to modify existing portions of the tag (such as the case of a VAR making a set of software look like it comes from a single entity).

4.1.41
tag provider
entity that creates (tag creator) or modifies (tag modifier) software identification tags for software packages

NOTE A tag provider can be part of the software provider organization, or can be a third party organization or the software consumer.

4.1.42
Uniform Resource Identifier
URI
compact sequence of characters that identifies an abstract or physical resource available on the Internet

NOTE The syntax used for URIs is defined in IETF RFC 3986.

4.1.43
valid
<XML file> software identification tag data follows the specified XSD definition and the software identification tag file is valid from an XML perspective

NOTE See also 4.1.44.

4.1.44
valid
<software identification tag> process used to ensure the data included in an installed software identification tag is correct

NOTE See also 4.1.43.

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

4.1.45
value-added reseller
company licensed to repackage and support existing products as combined software packages

4.1.46
version
unique string of number and letter values indicating a unique revision of an item

NOTE Versions are often referred to in software to identify revisions of software that provide unique functionality or fixes. A version typically has multiple parts with at least a major version indicating large changes in functionality or user interface changes and a minor version indicating smaller changes in functionality or user interface changes.

4.1.47
XML Schema Definition
XML based language that specifies a set of rules and structure for the creation of XML documents

NOTE XML documents follow all rules defined in an XSD definition in order to be considered a "valid" document.

4.2 Abbreviated terms

CI	configuration item
CMDB	configuration management database
GUID	globally unique identifier
IETF	Internet Engineering Task Force
MD5	message digest 5
regexp	regular expression
regid	registration identifier
SAM	software asset management
URI	uniform resource identifier
URL	uniform resource locator
VAR	value added reseller
W3C	World Wide Web Consortium
XML	Extensible Markup Language
XSD	XML Schema Definition

5 Alignment and rationalization with prior standards

5.1 Statement of alignment for this part of ISO/IEC 19770

The contents of this part of ISO/IEC 19770 are intended to complement and align with prior ISO/IEC standard publications.

5.2 Alignment with ISO/IEC 19770-1:2006 Information technology — Software asset management — Part 1: Processes

The following areas of ISO/IEC 19770-1:2006 are supported by this part of ISO/IEC 19770.

a) ISO/IEC 19770-1:2006, clause 3 stipulates terms and definitions relevant to that document.

   This part of ISO/IEC 19770 aligns with ISO/IEC 19770-1:2006, terms and definitions in clause 3 relevant to both parts have been reproduced here.

b) ISO/IEC 19770-1:2006, clause 4.4.2.2 stipulates: "Implementation of the software asset identification process will enable the organization to demonstrate that a) types of assets to be controlled and the information associated with them are formally defined. b) a register of stores and inventories exists,

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

clarifying which stores and types of information are held, with duplication allowed only if duplicate information can be traced back to the definitive source record."

This part of ISO/IEC 19770 affirms the necessity of formal definitions, stores and inventories for software configuration items. According to ISO/IEC 19770-1:2006, clause 4.4.2.2.a.2.i, software configuration items include "all platforms on which software can be installed or run." There is no explicit requirement for a "platform" element in 19770-2 because this part of ISO/IEC 19770 is focused on the identification of software found on computing devices and not on the requirements of a general software repository. It is expected that discovery tools collect platform information during their inventory process.

c) ISO/IEC 19770-1:2006, clause 4.4.2.2 stipulates: "Basic information required for all assets is i) Unique identifier ii) Name/description iii) Location iv) Custodianship (owner) v) Status (e.g., test/production status; development or build status) vi) Type (e.g., software, hardware, facility), vii) Version (where applicable)."

This part of ISO/IEC 19770 affirms these requirements for basic information. The location and custodianship of a software configuration item, however, are not included as values specified in this part of ISO/IEC 19770 as these are associated with the asset on which the software configuration item is discovered and not with the item itself.

The status of a software configuration item is defined by the Release values in the software identification tag. These values are optional and it is recommended that they are furnished alongside information pertaining to the sign off date and the operator who performed the process (8.4).

5.3 Alignment with ISO/IEC 20000-1:2005 Information technology – Service management – Part 1: Specification

a) ISO/IEC 20000-1:2005, clause 9.1 stipulates: "Changes to configuration items shall be traceable and auditable where appropriate, e.g. for changes and movements of software and hardware." This part of ISO/IEC 19770 affirms the usage of software identification tags for disclosure and definition of traceable and auditable information for software configuration items.

b) ISO/IEC 20000-1:2005, clause 9.1 stipulates: "Configuration control procedures shall ensure that the integrity of systems, services and service components are maintained."

This part of ISO/IEC 19770 upholds the necessity of configuration control procedures for integrity assurance purposes and therefore provides creators with the option to include a digital signature (6.1.11). The digital signature can be used to validate that specified mandatory element values have not been modified, this validation in turn allowing software providers or tag providers to authoritatively identify software identification tag tampering, or lack thereof.

c) ISO/IEC 20000-1:2005, clause 9.1 stipulates: "Master copies of digital configuration items shall be controlled in secure physical or electronic libraries and referenced to the configuration records, e.g. software, testing products, support documents."

This part of ISO/IEC 19770 that recommends software identification tags be included with respective software configuration items in the definitive software library.

d) ISO/IEC 20000-1:2005, clause 9.1 stipulates: "All configuration items shall be uniquely identifiable and recorded in a CMDB to which update access shall be strictly controlled."

This part of ISO/IEC 19770 aligns with the specification that all software configuration items be uniquely identifiable. A software configuration item is uniquely identifiable by product identifier, serial number, stock keeping unit, either of which can be related back to proof of license, purchase order and software configuration items stored in the CMDB (8.4.14, 8.4.20, 8.4.21).

The method by which a software identification tag is stored in the CMDB and referenced uniquely therein is not within the scope of this part of ISO/IEC 19770 (1.2).

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

e) ISO/IEC 20000-1:2005, clause 10.1 stipulates: "Release and distribution shall be designed and implemented so that the integrity of hardware and software is maintained during installation, handling, packaging and delivery."

This part of ISO/IEC 19770 affirms the necessity of integrity within the release process and therefore specifies that software identification tags fully conform to release processes detailed in ISO/IEC 20000-1:2005. This part of ISO/IEC 19770 recommends that optional elements pertaining to release details be included with software identification tags (8.4).

5.4 Alignment with ISO/IEC 20000-2:2005 Information technology — Service management — Part 2: Code of practice

a) ISO/IEC 20000-2:2005, clause 10.1.5 stipulates: "Release and distribution should be designed and implemented to: a) conform with the service provider's systems architecture, service management and infrastructure standards; b) ensure the integrity is maintained during build, installation, handling, packaging and delivery..."

This part of ISO/IEC 19770 affirms the importance of release and distribution conformance through the creation of the optional element "Release package" (8.4.17).

b) ISO/IEC 20000-2:2005, clause 10.1.6 stipulates: "The verification and acceptance processes should: a) verify that the controlled acceptance test environment matches the requirements of the target production environment; b) ensure that the release is created from versions under configuration management and installed in the acceptance test environment using the planned production process..."

This part of ISO/IEC 19770 affirms the importance of matching the controlled acceptance test environment to the requirements of the target production environment through the creation of the optional element "Release verification" (8.4.19).

c) ISO/IEC 20000-2:2005, clause 10.1.8 stipulates: "It is important that the release is delivered safely to its destination in its expected state."

This part of ISO/IEC 19770 upholds the need for efficient and secure release delivery by proposing the creation of the optional element "Release rollout" that allows an organization to validate who signed off on a software package as ready for production use and when the sign off occurred (8.4.18).

6 Implementation of software identification tagging processes

6.1 General requirements and guidance

6.1.1 Software identification tag overview

Annex A provides an overview of software identification tagging principles from a more conceptual perspective, to assist in understanding.

6.1.2 XML and XSD

The software identification tag shall be defined as an XML data structure. The XML Schema Definition (XSD) to be used shall be that defined in Annex G, or any (updated) version which may be downloaded from:

http://standards.iso.org/iso/19770/-2/2009/schema.xsd

Additional versions of the schema may be available with the version identifier of the schema included in the path to the schema. All prior versions of the schema shall be retained.

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

6.1.3 Unique registration ID (regid)

Software identification tags may be created by multiple different organizations and do not strictly require a centralized registration authority. Additionally, this part of ISO/IEC 19770 allows entities to create software identification tags for software configuration items they did not create (such as an organization creating software identification tags for their internal software discovery processes). To accommodate these requirements, this part of ISO/IEC 19770 will use a regid. The regid is based off of the iSCSI Qualified Name as defined in the IETF RFC 3720 – section 3.2.6.3.1 and the IETF RFC 3721 section 1.1 and provides a unique naming authority reference.

A regid can be created by any individual or organization that owns or has owned the registration for a domain name (as specified in IETF RFC 1034, section 3.5 and IETF RFC 1123, section 2.1). The domain name does not need to be active, nor does it need to resolve to an address. Domain names by themselves do not constitute a unique identifier since domains because they can expire and/or be acquired by other entities this means a regid must also include a date that the domain registration was owned by the entity. Finally, for entities that wish to further sub-divide unique naming sub-entities, an optional suffix is provided for the regid that may be used, for example, to provide large software publishers means to allow each of their business units to manage their own software identification tags independently.

The regid name shall consist of the following:

- The string "regid" – this qualifies the element as a registration id for software identification tags.
- A dot '.'
- A date code in YYYY-MM format. This date shall be a date during which the naming entity owned the domain. This date should be the first month in which the domain name was owned by this naming entity at 00:01 GMT of the first day of the month. This date code uses the Gregorian calendar and must include all four digits of the year and both digits of the month (where January = 01 and December = 12). The dash must be included.
- A dot '.'
- The reversed domain name of the naming entity (person or organization) creating a software identification tag
- An optional string that specifies sub-entities that may be their own unique naming authorities. This is specified by:
    - A comma ','
    - With the exception of the comma prefix, the owner of the domain name can assign text following the reversed domain name as desired as long as all characters are valid for use in filenames on any platforms the tag will be installed on. It is the responsibility of the naming entity to ensure that each sub-entity reference is unique within their organization.

An example of regids created by entities owning example.com or example.net looks as follows:

Table 1 — Examples of regid values

```
                    Naming      Additional
Type   Date         Auth        "example.com" naming authority
+---+ +-----+ +---------+ +--------------------------------+
|   | |     | |         | |                                |

regid.1995-09.com.example,AccountingSystems
regid.1995-09.com.example
regid.1995-09.net.example,WordProcessing
```

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

6.1.4 Software identification tag extension and location for installation

Software identification tag files shall include the ".swidtag" file extension in their names for the recognition purposes of platforms (4.1.7) and discovery tools.

Each platform provider (4.1.18), e.g. vendor of an operating system, should specify where software identification tags are to be located. Windows® installations, for instance, may have a Windows Management Instrumentation™ value specifying location; Linux™ distributions may use a Red Hat Package Manager value. If the platform provider does specify a location, then the software identification tag shall be installed in this location.

In the absence of specifications from the platform provider, software identification tags should be installed in commonly known shared locations that are used for collecting commonly used system information. The following examples provide information on which shared locations are expected to be used for various platforms, in the absence of an alternative specification from the platform provider.

Table 2 — Examples of tag locations on different platforms

Apple Macintosh™ OS:X™ Leopard	<root>/Library/Application Support/**<software creator regid>**
Apple Macintosh™ OS X™ pre-Leopard NOTE software identification tags should be included in the application directory by default for all operating systems (see below). Pre-Leopard OS X systems should also use this location as the default location.	Application Directory/<program.app package>/contents
UNIX® and Linux™	usr/share/**<software creator regid>**
Windows® NT	C:\Winnt\All Users\Application Data**<software creator regid>**
Windows® 2000 Professional Windows Server® 2000 Windows® XP Windows Server® 2003	%AllUsersProfile%\Application Data**<software creator regid>**
Microsoft Vista® Microsoft Server® 2008	%Program Data%**<software creator regid>**

The platform provider may provide access to the software identification tag using methods that are independent from file access. For example, Microsoft Vista® includes 4 APIs that may be used to manage a software identification tag repository. These API's are:

Table 3 — Microsoft Vista® APIs for software identification tag management

SLGetInstalledSAMLicenseApplications	Retrieves a list of applications that have a software identification tag installed by using SLInstallSAMLicense
SLGetSAMLicense	Gets information about a specific software identification tag installed by using SLInstallSAMLicense
SLUninstallSAMLicense	Removes a software identification tag for a specified application
SLInstallSAMLicense	Adds a software identification tag to the Microsoft Vista® repository

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

In addition a copy of the software identification tag shall also be installed in the top level directory of the application itself. This allows the software identification tag to be discovered even if a removable storage device (such as a USB hard disk) is moved from one system to another (7.5). Should the same software be installed in two different locations for the same set of end-users, the expectation is that there would still be two (or more) instances of the software identification tag in the common system location specified above as well as two (or more) other tags that are located in the root directories of the installation directories. In this case, a software uninstaller needs to be aware of the multiple installations at uninstall time in order to ensure that a software identification tag is not removed from the common system location until all installations are uninstalled.

NOTE SAM tool providers should be aware that there will be cases where a software identification tag can be found in the common system location as well as in one or more top level directories of a software package installation. Using information in the installation_details element, SAM tools can associate tags located in the common system location and installation root directories with their corresponding installations. Rules need to be included in the SAM tools to deal with various permutations of tags which are found. In cases where the tag is found in the software package installation directory, and not in the common system location, additional rules are required to identify if software is located on removable storage media that may have been moved to another system. If a tag is found in the common system location as well as multiple tags found in installation directories, rules may need to be applied as appropriate for the organizational policies. Appropriate reporting and action can then be taken by the SAM practitioner.

The goal of a SAM tool should be to make it as easy as possible for a SAM practitioner to manage exceptions to organizational policy, so recognizing some of these issues and reporting on them based on the policies specified by the practitioner make overall SAM implementations significantly easier to manage.

6.1.5 Unique identifiers

For the purpose of uniqueness, there are two elements that, combined, shall create a globally unique ID called the software_id. These elements are

a) tag_creator_regid

b) unique_id that may be either a GUID, or any reference unique for the tag_creator_regid. The unique_id shall follow the restrictions for URI character use as specified in IETF RFC 3986, section 2, Characters.

The benefits of implementing a unique identifier during software identification tag creation include, but are not limited to, facilitation of the following:

a) Identification of parent-child relationships.

b) Explicit definition of dependencies and recognition of dependent software.

c) Identification of upgrade software and allowed upgrade packages.

d) Reference to identifying software identification tags from within software configuration items.

6.1.6 Unique software identification tag file name – distribution

When a software identification tag is created for distribution on installation media, it is not possible to provide additional installation-specific unique ids as described in section 6.1.7 since the tag has not been installed on any computing device as yet. In this case, the tag file is part of the master image for the installation process and shall align with the following structure:

<tag_creator_regid>_<software_id.unique_id>.swidtag

Following this structure provides a unique filename that can ship with the software.

A software identification tag on the installation media would thus typically have a filename such as:

regid.1986-12.com.adobe_fc3cc419-b5a1-9f16-ed203e537c40.swidtag

The installation routine shall then follow the process identified in section 6.1.7 when the tag is installed.

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

6.1.7 Unique software identification tag file name - installed

When software is installed on a computing device, the name of a software identification tag shall be unique at least to the specific code to be installed, and align with the following structure:

<tag_creator_regid>_<software_id.unique_id>_<unique_sequence_id>.swidtag

The unique_sequence_id is optional and shall be used to ensure that every software identification tag installed on a computing device has a unique filename. This unique_sequence_id may be a simple numerical sequence, or it may be created using an algorithm chosen by the organization that installs the software identification tag. The choice of how to create the unique_sequence_id is up to the organization installing the software identification tag, but the methodology used shall ensure that the filename is unique for a specific machine and/or virtual environment. The initial portion of a software identification tag filename would look as follows:

regid.1986-12.com.adobe_fc3cc419-b5a1-9f16-ed203e537c40

This would be followed by the unique_sequence_id. The algorithm to specify the unique_sequence_id may include some of the following considerations:

1. For basic tag filenames, the unique_sequence_id may simply be a sequential value that, if during installation of a software identification tag, a tag with the same name already exists, the sequential number is incremented. In this instance, the numerical sequence would start at a specified reference number and increment as other software identification tags for the same software ID are installed (this would happen if and only if the software allows multiple installations on a particular computing device). Note that the installation_instance within the optional installation_details element could be used as part of the filename in this way.

   An example of this software identification tag file name (for the first installation) is:

   regid.1986-12.com.adobe_fc3cc419-b5a1-9f16-ed203e537c40_1.swidtag

   tag_creator_regid | unique_id | numerical sequence #

2. The preferred algorithm for the unique_sequence_id should include details specific to the computing device and/or media on which the software is installed (this may include device serial number, NIC MAC addresses, hard disk serial number, or other unique reference items) and/or details about the virtual environment where the software has been installed. Note that the installation_target_id within the optional installation_details element may be used as part of the filename in this way. Having device -specific details facilitates the tracking of software identification tags that may be installed on removable or shared media. Note that this algorithm may also require an additional unique sequence number to ensure that multiple installations on the same machine do not create the same filename. The installation_instance within the optional installation_details element could be used as part of the filename in this way. An example of this software identification tag file name is:

   regid.1986-12.com.adobe_fc3cc419-b5a1-9f16-ed203e537c40_001c26f781fb_1.swidtag

   tag_creator_regid | unique_id | installation _ target_id | numerical sequence #

   Regardless of the algorithm chosen, the software identification tag filename shall be created by the installation routine that installs the tag and the name must not conflict with any existing filename. Additionally, the characters used in the filename shall not exceed 254 characters (or less if the targeted platform for the software requires shorter file names). Finally, the characters used in the filename shall meet all specific criteria required for the file systems that the tag is targeted for.

The .swidtag file extension shall be used for all software identification tags. This naming scheme allows for multiple software identification tags to be applied to the same product title, thereby providing support for

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

upgrades. It also allows for unique software identification tags to be created by organizations other than the original software creator, such as for legacy software.

6.1.8 Consistency among data values

One challenge for software asset management is the fact that marketing names used by software providers change frequently. Since the software identification tag will not typically be seen by the end-user, this tag can provide a consistent and reliable source of information to enable effective inventory and software entitlement reconciliation. Keeping values consistent from one version to another or across product lines is a large part of the value software identification tags provide the SAM practitioner.

Numerous elements in the software identification tag should remain consistent from tag to tag to optimize the overall software asset management process. It is recommended, for example, that tag providers maintain consistency in the following elements:

a) Software creator identity (8.3.4) – this element should remain consistent across all software packages created by a specific company.

b) Software licensor identity (8.3.5) – this element should remain consistent across all software packages licensed by a specific company.

c) Tag creator identity (8.3.7) – this element should remain consistent across all software identification tags created by a specific company.

d) License and channel information (8.4.8) - this optional element structure should remain consistent across product lines and it is recommended that the values are consistent across all software created by a specific company. Note that data in the ‘license and channel information’ elements of a software identification tag do not specify software license or software entitlement information – instead, data in these elements provides guidance to a SAM practitioner that will help them determine and perhaps automate software license reconciliation procedures.

e) Product category (8.4.12) – this optional element should remain consistent for a particular product unless the product adds or removes functionality to make it obvious that it belongs in a different category.

f) Product identifier (8.4.14) – this optional element should remain consistent especially for products that have maintenance agreements that provide for upgrade rights over a period of time. his element is used to identify a specific product from release to release – this element is not a product name, it is simply an identifier so the relationship for upgrade purposes can be done automatically.

6.1.9 Software identification tag discovery

Software is generally created as a gold master copy by software providers, then copied and distributed through different channels. Depending on the software provider's requirements, the software identification tag may be incorporated directly into the gold master, or it may be created by the installer, or even the software package itself. The primary requirement is that the software identification tag for the package shall be discoverable on the machine, media and/or virtual environment on which the package is installed. For more details, refer to the annex detailing guidance for software providers (Annex B).

6.1.10 Languages

Acknowledging that many software creators produce software with specific builds that are dependent on language while many others produce software with one build that implements add-on "language packs," this part of ISO/IEC 19770 does not require software identification tags to recognize different language versions of the same product. Consistent categorization through the use of the supported languages element, is however, strongly encouraged (8.4.24).

For encoding purposes, the use of utf-8 is the suggested methodology for software identification tags created based on this part of ISO/IEC 19770 (see http://www.w3.org/International/O-charset).

Documents
Document Sharing Hub

© ISO/IEC 2009 – All rights reserved

Full document details: BS ISO/IEC 19770-2:2009

6.1.11 Ownership of elements within software identification tags

Because of the way that different entities may add or modify elements within the software identification tag, it may be necessary for any one of these entities to identify which elements it has created and/or modified. This capability is provided primarily via the elements_owner element, making use of the internal element ID as described in 6.1.12.

Alternatively, ownership of individual elements may be identified through the use of digital signatures as described in 6.1.13. However, the use of digital signatures does not directly identify an owner in the same way as the elements_owner does, so the use of the elements_owner element is still recommended even when digital signatures are used.

6.1.12 Internal element ID

There is a requirement to be able to cross-refer internally within the software identification tag to other individual elements within the software identification tag. One obvious situation where this is required is in the ability to identify the elements which have been created and/or modified by a specified tag creator or modifier, as listed in the elements_owner element. This cross-reference capability is provided by assigning internal element IDs to the individual elements which may then be used to cross-refer to those elements from other elements.

Internal element IDs are used for intra-tag reference as well as for inter-tag references where a specific element from another tag must be identified (this may happen in the case where a digital signature is provided in a secondary file). These elements IDs must be unique within a software identification tag, but need not be unique between tags. For software identification tags, ID attributes are typically used to identify specific ownership of elements. The tag creator and modifier may use any defined methodology to specify and use ID attributes. However, it is useful if a default convention is provided and used when possible. he default convention suggested for this part of ISO/IEC 19770 is:

a) Every XML ID must start with an alphabetic character. The convention for this part of ISO/IEC 19770 is for the ID to start with the letter 'e' standing for element.

b) Each top level element should utilize the same value as the paragraph defining that element with each component of the paragraph denoted with an underscore. This results in the following attribute for the element product_version:

```
<swid:product_version ID="e8_3_3">
```

c) Every element may have sub-elements. In these cases, for each sub element, the ID should start with the top level paragraph identifier (as noted above), followed by "sub[number of sub element]". This results in the following IDs specified for the structure under product_version:

```
<swid:name ID="e8_3_3sub1">10.2</swid:name>
<swid:numeric ID="e8_3_3sub2">
  <swid:major ID="e8_3_3sub2sub1">10</swid:major>
  <swid:minor ID="e8_3_3sub2sub2">2</swid:minor>
  <swid:build ID="e8_3_3sub2sub3">0</swid:build>
  <swid:review ID="e8_3_3sub2sub4">0</swid:review>
</swid:numeric>
```

d) For values that may have multiple entries specified, such as abstract, the ID should start with the top level paragraph identifier (as noted above), followed by a sequence number as specified by "seq[number of instance]". This results in the following IDs specified for the structure under abstract:

```
<swid:abstract lang="en" ID="e8_4_1_seq1">This is the abstract written in English</swid:abstract>
<swid:abstract lang="fr" ID="e8_4_1_seq2">This is the abstract written in French</swid:abstract>
```


© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

Following these recommendations for the ID references used by elements_owner provides a consistent approach to specifying IDs that may readily be understood by an engineer who may be reading the tag as well as being managed effectively through automated approaches.

6.1.13 Authenticity of software identification tags

Often, there will be a desire, or need to prove the authenticity of a software identification tag. For example, during an audit, a software vendor may want to validate that the software identification tag collected during a discovery process has not had key elements of the tag altered. Authentication is supported by allowing digital signatures within the software identification tag.

There is an existing recommendation published by the W3C that addresses the need to provide digital signatures in an XML document. The recommendation is "XML-Signature Syntax and Processing (Second Edition) – 10 June, 2008" found at the following location:

http://www.w3.org/TR/xmldsig-core/

NOTE W3C does not reference official documents as "standards". Instead, the officially supported documents are referred to as "recommendations". The reader should be aware that W3C follows a well defined process to get a document to a recommendation level and that W3C recommendations should be considered as authoritative documents.

The W3C recommendation provides integrity message authentication as well as signer authentication services for data of any type.

This part of ISO/IEC 19770 does not define the process for applying digital signatures to a software identification tag since the W3C recommendation already does that.

Signatures are not a mandatory part of the software identification tag, and can be used as required by any tag creator or modifier to ensure that sections of a tag are not modified and/or to provide authentication of the signer. If signatures are required for the software identification tag, they shall follow the W3C recommendation defining the XML Signature Syntax (http://www.w3.org/TR/xmldsig-core/).

NOTE Software identification tags will generally not require XML Advanced Electronic Signatures (XAdes), so this W3C recommendation is not referenced in this part of ISO/IEC 19770.

6.1.14 Standardization of XSD definition

There are a number of standardized types used in the software identification tag including specific date/time entries that shall follow a specified formats as specified in the W3C recommendation titled, "*XML Schema Part 2: Datatypes Second Edition*". Details for these data types can be found at the following location:

http://www.w3.org/TR/xmlschema-2/

By using specific types as specified by the above W3C recommendation, software identification tags can go through an automated validation step that allows a much more consistent structure to the data provided.

NOTE A few elements defined in this part of ISO/IEC 19770 provide for the use of regular expressions. The syntax that is supported for these regular expressions is also defined in the W3C recommendation listed above.

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

6.2 Software identification tagging life cycle: operational breakdown

6.2.1 Introduction

The following diagram shows the software identification tag lifecycle as it progresses from a software creator (or tag creator) to a software consumer organization.

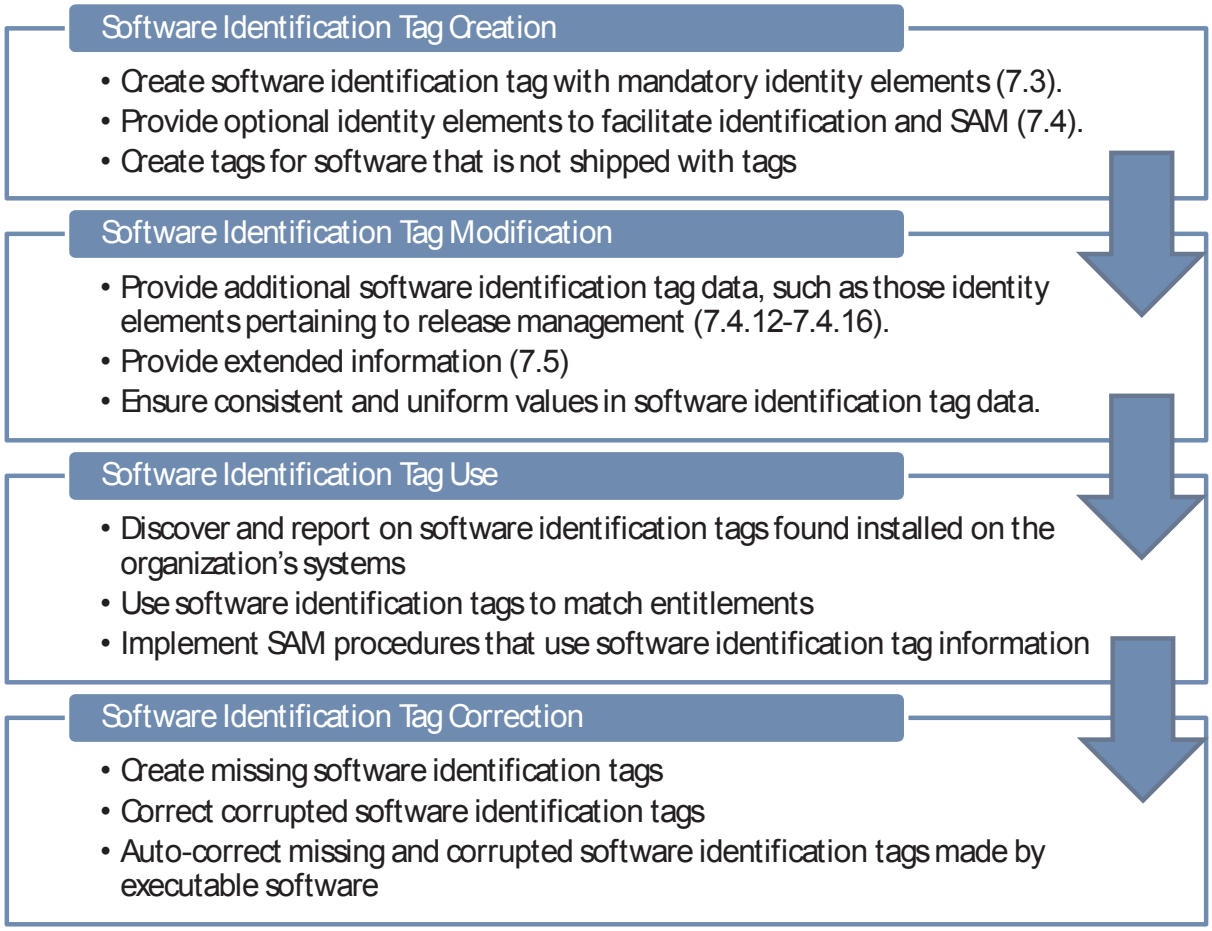

Figure 1 — Software identification tag lifecycle

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

6.2.2 Software identification tag creation

The software identification tag creator will often be the same as the software creator. As software is built, the software creator develops a corresponding software identification tag to identify the software package.

Example software identification tag creators include:

a) *Software manufacturers.* A software identification tag may be created when a software manufacturer develops a particular software configuration item. Subsequently, the software identification tag and software installation routine are shipped together. Software may be dispatched directly to software consumers, publishers or both. Arrangements as to who creates the software identification tag and who defines the software configuration item's origins may be agreed to between the various parties. It is possible that this process may have mixed ownership as it depends on the channel used for software distribution. Software manufacturers will likely use digital signatures regularly within software.

b) *Software publishers.* When software is developed by one organization and then published by another, during the packaging process the software publisher will create a software identification tag to include as part of the product installation process.

c) *Line of business application developers.* Line-of-business application developers will also create software identification tags to include as part of the product installation process.

d) *Distributors, repackagers, value-added resellers and other tag modifying organizations.* Organizations that distribute software that does not include standardized software identification tags may want to add them in order to accommodate the needs of their software consumers. When a software package does not include a software identification tag, SAM practitioners are encouraged to create and include their own software identification tag to optimize SAM processes.

Software identification tag creators may also be third party organizations that are not directly related to the software creator. This may be the case for legacy software that does not have software identification tags, or for software that is developed by an organization that chooses not to include software identification tags with their products. In these cases, the tag creator shall define element values in the software identification tag that indicate that they are not the creator of the software (such as the tag_creator_regid in the software_id element). These cases will be evident because the tag_creator_regid will be different from the software_creator_regid.

6.2.3 Software identification tag modification

Individuals or companies that alter software identification tags and/or add supplemental information to software identification tags are considered tag modifiers. This group may include software aggregators, VAR's and may also include groups within an organization that manage software release processes.

Example software identification tag modifiers may include:

a) Distributors

b) Resellers

c) Value-added resellers

d) Republishers

e) Packagers

f) Discovery tool providers

g) Deployment tool providers

h) Release managers

© ISO/IEC 2009 – All rights reserved

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009

6.2.4 Software identification tag use

Although software consumers are the ultimate beneficiaries of standardized software identification tag content, discovery tools and operating systems are the primary systems that use the data. They read software identification tag data to gather information about a given set of software configuration items. When a software configuration item is deposited onto a particular computing device, a well-formed software identification tag provides authoritative identification of the specific software installed, the tag creator and all third-party providers who may have modified the software. This information is invaluable to the SAM practitioner.

Example software identification tag end-users include:

a) SAM owners

b) IT support professionals

c) Owners of the software configuration item

6.2.5 Software identification tag correction

Although software identification tags should be managed along with all other components associated with a specific software installation, there will be times when a tag may be removed, deleted or corrupted – either on purpose or accidentally.

In the case where a software identification tag becomes corrupt, there are optional methods available for a software application to validate that the software identification tag is up-to-date and correct and/or potentially to automatically self-heal the software identification tag. These methods help to ensure a higher level of integrity of the data collected by discovery tools.

In the case where a product installation does not include a software identification tag, release managers may choose to create a software identification tag to provide with the software to ease the SAM reconciliation process.

In all cases, there should be methods available for a discovery agent or service to cross check information in a manner that provides assurance about the accuracy of the data. For example, a discovery agent may collect software identification tags and all executable filenames from a system. Software identification tags have the ability to specify all the files that are associated with a particular piece of software. Using an appropriate algorithm, a SAM discovery tool can quickly validate that a system with a specific software identification tag also includes the necessary files that are associated with that title. The reconciliation engine can then filter out the known application filenames from the collected list of executable filenames. Any filenames that remain in the list at the end of this process would be considered exceptions and may require that a SAM practitioner investigates these files further.

7 Platform requirements and guidance

7.1 Types of platforms

For the purposes of this part of ISO/IEC 19770, the term platform refers to a computer or hardware device and/or associated operating system, or virtual environment, on which software can be installed or run (4.1.17). The Linux™ operating system, for example, is used on a wide variety of hardware, from cell phone devices to mainframe computers, and each variation can be considered a separate platform for the purposes of this part of ISO/IEC 19770. Additional example platforms include, but are not limited to:

a) Redhat™ Linux™ Enterprise 4.0 Intel x86

b) Novell SuSE™ Linux™ 10.2 Intel x64

c) Macintosh™ OS 10.4 Intel x64

© ISO/IEC 2009 – All rights reserved
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 19770-2:2009