8.4.12 Product category ('product_category')

<table>
<tr><td>XML Tag</td><td colspan="3">product_category</td></tr>
<tr><td>Type</td><td colspan="3">Complex type</td></tr>
<tr><td>Definition</td><td colspan="3">Means by which product titles are classified by high-level function. A standardized list of categories/groups is provided by the United Nations Standard Products and Services Code: UNSPSC (for more information see http://www.unspsc.org/), COMMODITY listing number 43230000). UNSPSC codes found in the section numbered 43230000 of the specification are where the bulk of commonly used software categories will be found. Product categorization shall be done using the UNSPSC codes.

This element may occur zero to one time in the software identification tag.</td></tr>
<tr><td rowspan="8">Data Structure</td><td>XML tag</td><td>Type</td><td>Definition</td></tr>
<tr><td>UNSPSC_ver</td><td>XML character string

One entry</td><td>Version number of the UNSPSC code set used. The version is not required to use the code, however, if a tool uses the version to provide additional functionality (such as providing various names in one of 9 other languages), the version will be needed by the tool.

An example of the format of the UNSPSC versions is 10.0501.</td></tr>
<tr><td>segment_title</td><td>XML character string

One entry</td><td>Name of the segment the product belongs to</td></tr>
<tr><td>family_title</td><td>XML character string

One entry</td><td>Name enabling recognition of the family of the product</td></tr>
<tr><td>class_title</td><td>XML character string

One entry</td><td>Name of the class</td></tr>
<tr><td>commodity_title</td><td>XML character string

One entry</td><td>Name of the commodity</td></tr>
<tr><td>code</td><td>Numerical value with 8 digits

One entry</td><td>Codes shall be specified as defined in the UNSPSC code list.</td></tr>
<tr><td colspan="3"></td></tr>
<tr><td>Example</td><td colspan="3"><category>
<UNSPSC_ver>10.0501</UNSPSC_ver>
<segment_title>Information Technology Broadcasting and Telecommunications</segment_title>
<family_title>software</family_title>
<class_title>Finance accounting and enterprise resource planning ERP software</class_title>
<commodity_title>Enterprise resource planning ERP software</commodity_title>
<code>43231602</code>
</category></td></tr>
</table>

8.4.13 Product family ('product_family')

XML Tag	product_family
Type	XML character string
Definition	Product family provides an element software publishers and software licensors can use to group related software products together for SAM practitioner reports. An example of the type of product that would use this element is a backup tool where the backup services, server backup and client backup portions for the tool are sold as independent products. In this case, if all products have the same product_family defined, a SAM tool can automatically group discovered software identification tag data appropriately as shown below: Example Backup Utility Backup Server - 20 installations discovered Server Backup Utility – 240 installations discovered Client Backup Utility – 10,240 installations discovered
Example	<product_family>Example Backup Utility</product_family>

8.4.14 Product identifier ('product_id')

XML Tag	product_id
Type	XML character string
Definition	Identification of the product. It is independent from its version. Product_id should be a unique reference, but this can be unique within the software manufacture and does not need to be a globally unique ID. It is recommended that the Product ID not be the product name, or other marketing term as these often change from release to release. Instead the product_id should be an identifier that can follow products through their lifecycle without requiring marketing changes. Product_id is used to define a lineage between products for identification of allowed upgrades. This value may or may not be used by a software entitlement. If a software entitlement specifies that a product may allow upgrades during a certain period of time, the software entitlement document cannot know which future product names or product versions can be applied and will become available during that time. The product_id allows a software entitlement document to specify that a specific version of the product is entitled to be installed initially, and any updated products that have the same product_id are also entitled to be installed as long as the release_date falls within the range provided in the software entitlement. NOTE There may be more than one entry for product_id. This may happen in the case where a creator comes out with a new product and allows end-users or software consumers using different older products to upgrade to the new one. For example: Product A product_id = 1234XYZ Product B product_id = ABCDPDQ Product C (this product allows maintenance upgrades from Product A or Product B) product_id = 9876HJK <- this is the new product ID for Product C… product_id = 1234XYZ

	product_id = ABCDPDQ If later releases of Product C only wanted to allow maintenance upgrades from earlier versions of Product C, the product_id would only include the new ID for that product – 9876HJK. This element may occur zero to unlimited times in the software identification tag
Example	<product_id>fc3cc419-b5a1-9f16-ed203e537c40</product_id>

8.4.15 Release date ('release_date')

XML Tag	release_date
Type	XML dateTime type
Definition	This tag will typically be used by a software consumer organization as part of an ITIL release process. Date software configuration item was released for installation. The software configuration item should use a single date of release in order to facilitate reconciliation. This element may occur zero to one time in the software identification tag.
Example	<release_date>2008-01-21T12:00:00</release_date>

8.4.16 Release identifier ('release_id')

XML Tag	release_id
Type	XML character string
Definition	This tag will typically be used by a software consumer organization as part of an ITIL release process. Data used in reconciliation to identify release package attributes upon installation and associated software entitlements. Entries for this element shall be kept consistent across all software identification tags for any given software configuration item. This element may occur zero to one time in the software identification tag.
Example	<release_id>COE-Base-Ver 8, 2008-01-21</release_id>

8.4.17 Release package ('release_package')

<table>
<tr><td>XML Tag</td><td colspan="3">release_package</td></tr>
<tr><td>Type</td><td colspan="3">Complex type</td></tr>
<tr><td>Definition</td><td colspan="3">This tag will typically be used by a software consumer organization as part of an ITIL release process.

Validation information that a release package has been built to conform to the service provider's systems architecture, service management and infrastructure specifications.

NOTE End-use software packages will almost always be customized to the needs of the service provider, with specific installation options and/or combinations of software bundling specified. (Release software identification tags are completely independent of any external software provider software identification tag).

This element may occur zero to one time in the software identification tag.</td></tr>
<tr><td rowspan="4">Data Structure</td><td>XML tag</td><td>Type</td><td>Definition</td></tr>
<tr><td>Sign_off</td><td>XML character string

One entry</td><td>This entry indicates the person who authorized that the software was packaged properly and is ready to go into a testing phase.</td></tr>
<tr><td>Sign_off_date</td><td>XML dateTime type

One entry</td><td>This entry indicates the date the software package was signed off.</td></tr>
<tr><td>By</td><td>XML character string

One entry</td><td>This entry indicates the software developer who created the package. This information may be used if questions come up during the testing phase.</td></tr>
<tr><td>Example</td><td colspan="3"><release_package>
  <sign_off>Jane Doe</sign_off>
  <sign_off_date>2008-01-10T12:00:00</sign_off_date>
  <by>John Doe</by>
</release_package></td></tr>
</table>

8.4.18 Release rollout ('release_rollout')

XML Tag	release_rollout
Type	Complex type
Definition	Validation information relevant to who signed off a release package as ready for production use and when the sign off occurred. This element may occur zero to one time in the software identification tag.

Data Structure	XML tag	Type	Definition
	Sign_off	XML character string One entry	This entry indicates the person who authorized that the software was properly tested in a production pilot and is ready to go into a production use.
	Sign_off _date	XML dateTime type One entry	This entry indicates the date the software pilot was signed off.
	By	XML character string One entry	This entry indicates the SAM practitioner who managed the pilot testing phase. This information may be used if questions come up once the software is in production.
Example	<release_rollout> <sign_off>Mary Jane</sign_off> <sign_off_date>2008-01-16T12:00:00</sign_off_date> <by>John Smith</by> </release_rollout>		

8.4.19 Release verification ('release_verification')

XML Tag	release_verification		
Type	Complex type		
Definition	Validation information that a release package has been verified against a testing environment that matches the requirements of the target production environment. This element may occur zero to one time in the software identification tag.		
Data Structure	**XML tag**	**Type**	**Definition**
	Sign_off	XML character string One entry	This entry indicates the person who authorized that the software was properly tested in a controlled environment and is ready to go into a pilot testing.
	Sign_off_date	XML dateTime type One entry	This entry indicates the date the software testing was signed off.
	By	XML character string One entry	This entry indicates the SAM practitioner who managed the controlled testing phase. This information may be used if questions come up once the software is in the pilot testing phase.
Example	<release_verification> <sign_off>Jane Smith</sign_off> <sign_off_date>2008-01-14T12:00:00</sign_off_date> <by>Doug Johnson</by> </release_verification>		

8.4.20 Serial number ('serial_number')

XML Tag	serial_number
Type	XML character string
Definition	Unique identifying number; may be represented as a combination of numbers, letters or symbols. Serial Number is a commonly used unique number assigned for identification of a particular title and purchase. In the case of software identification tags, the unique_id becomes the primary unique key, but many organizations may still want to use the serial number where it is available. NOTE 1 The serial number may be put through a one way hash that obfuscates the actual serial number – this is still useful to the SAM practitioner – especially if the same reference serial number is included on the purchase order, invoice or other details provided by the distributor to the software consumer. NOTE 2 If the tag creator chooses not to provide a serial number, they may choose to provide some other referencable data value that may be used to associate information in purchase orders. This allows a tag creator to assist SAM providers in finding entitlement information. This element may occur zero to one time in the software identification tag.
Example	<serial_number>1088-9015-2034-4567</serial_number> **Or** <serial_number>10PQR28FTQN2008</serial_number>

8.4.21 SKU ('sku')

XML Tag	sku
Type	XML character string
Definition	A Stock Keeping Unit (SKU) is a unique identifying number for a software provider. The SKU may be represented as a combination of numbers, letters or symbols. SKU is a commonly used unique number assigned for identification of a particular title and purchase. In the case of software identification tags, the unique_id becomes the primary unique key, but many organizations may still want to have direct access to the SKU value. NOTE If the tag creator chooses not to provide a SKU, they may choose to provide some other referencable data value that may be used to associate information in purchase orders. This allows a tag creator to assist SAM providers in finding entitlement information. This element may occur zero to one time in the software identification tag.
Example	<sku>065-04940</sku>

8.4.22 Software creator alias ('software_creator_alias')

<table>
<tr><td>XML Tag</td><td colspan="3">software_creator_alias</td></tr>
<tr><td>Type</td><td colspan="3">Complex type – EntityDataComplexType</td></tr>
<tr><td>Definition</td><td colspan="3">Provides additional software creator information enabling SAM practitioners and SAM tool providers to identify previous entities who were related to the creation of the software identified in the tag. Though not strictly required for software discovery purposes, this entry will ease the burden of a SAM practitioner by providing them with previous software creator details which can be used to more easily find an older software entitlement. This is especially important in the case where an upgrade is allowed from a previous software provider's version of a product to the current provider's version.

This element may occur zero to one time in the software identification tag.</td></tr>
<tr><td rowspan="2">Data Structure</td><td>XML tag</td><td>Type</td><td>Definition</td></tr>
<tr><td>alias</td><td>Complextype - AliasDetailsComplexType

Zero to unlimited entries</td><td>Details of previous creators who may have a relationship to the software title identified by the software identification tag.</td></tr>
<tr><td>Example</td><td colspan="3">The following example is appropriate for a Macrovision product that was purchased by and is now owned by Adobe®

<software_creator_alias>
<alias>
<alias_name>Macrovision</alias_name>
<alias_regid>regid.1998-02.com.macrovision</alias_regid>
</alias>
</software_creator_alias>

Or, if the regid of the alias entity is unknown:

<software_creator_alias>
<alias>
<alias_name>Macrovision</alias_name>
<alias_regid>unknown</alias_regid>
</alias>
</software_creator_alias></td></tr>
</table>

8.4.23 Software licensor alias ('software_licensor_alias')

<table>
<tr><td>XML Tag</td><td colspan="3">software_licensor_alias</td></tr>
<tr><td>Type</td><td colspan="3">Complex type – EntityDataComplexType</td></tr>
<tr><td>Definition</td><td colspan="3">Provides additional software licensor information enabling SAM practitioners and SAM tool providers to identify previous entities who were related to the licensing of the software identified in the tag. Though not strictly required for software discovery purposes, this entry will ease the burden of a SAM practitioner by providing them with previous software licensor details which can be used to more easily find an older software entitlement. This is especially important in the case where an upgrade is allowed from a previous software provider's version of a product to the current provider's version.

This element may occur zero to one time in the software identification tag.</td></tr>
<tr><td rowspan="2">Data Structure</td><td>XML tag</td><td>Type</td><td>Definition</td></tr>
<tr><td>alias</td><td>Complextype - AliasDetailsComplexType

Zero to unlimited entries</td><td>Details of previous licensors who may have a relationship to the software title identified by the software identification tag.</td></tr>
<tr><td>Example</td><td colspan="3"><software_licensor_alias>
<alias>
<alias_name>Adobe Systems</alias_name>
<alias_regid>regid.1986-12.com.adobe</alias_regid>
</alias>
</software_licensor_alias>

Or, if the regid of the alias entity is unknown:

<software_licensor_alias>
<alias>
<alias_name>Adobe Systems</alias_name>
<alias_regid>unknown</alias_regid>
</alias>
</software_licensor_alias></td></tr>
</table>

8.4.24 Supported languages ('supported_languages')

XML Tag	supported_languages		
Type	Languages as specified in IETF RFC 4646		
Definition	Languages that the program interface presents to the user. Languages shall be specified as defined in IETF RFC 4646. This element may occur zero to one time in the software identification tag.		
Data Structure	**XML tag**	**Type**	**Definition**
	Language	XML character string One to unlimited entries	Languages supported by this software package. Language may occur multiple times. Specification of the language shall be specified as defined by IETF RFC 4646 (see http://www.ietf.org/rfc/rfc4646.txt) and the process used for matching language tags is specified in IETF RFC 4647 (see http://www.ietf.org/rfc/rfc4647.txt).
Example	<supported_languages> <language>en</language> <language>fr</language> </supported_languages>		

8.4.25 Tag creator alias ('tag_creator_alias')

XML Tag	tag_creator_alias		
Type	Complex type – EntityDataComplexType		
Definition	Provides additional tag creator information enabling SAM practitioners and SAM tool providers to identify previous entities who were related to the creation of the software identification tag. Though not strictly required for software discovery purposes, this entry will ease the burden of a SAM practitioner by providing them with previous tag creator details which can be used to more easily find an older software entitlement. This element may occur zero to one time in the software identification tag.		
Data Structure	**XML tag**	**Type**	**Definition**
	alias	Complextype - AliasDetailsComplexType Zero to unlimited entries	Details of previous tag creators who may have a relationship to the software title identified by the software identification tag.
Example	The following example is appropriate for a Macrovision product that was purchased by and is now owned by Adobe® <tag_creator_alias> <alias> <alias_name>Macrovision</alias_name> <alias_regid>regid.1998-02.com.macrovision</alias_regid>		

	</alias> </tag_creator_alias> **Or**, if the regid of the alias entity is unknown: <tag_creator_alias> <alias> <alias_name>Macrovision</alias_name> <alias_regid>unknown</alias_regid> </alias> </tag_creator_alias>

8.4.26 Tag creator copyright ('tag_creator_copyright')

XML Tag	tag_creator_copyright		
Type	XML character string		
Definition	This element is provided in order to enable the tag creator the chance to specify the copyright for this particular tag. It is expected that software creators will allow their tag to be collected and distributed as long as creator-specified contents of the tag are not modified. This allows SAM tool providers and others to access and use software identification tags easily within their tools. An independent 3rd party that creates tags may put more limitations on the use and/or redistribution of the software identification tag data. See Annex F for more details on copyright information. The abstract element may occur more than once in a software identification tag, but shall only occur once for each language specified. If language is not specified, it is assumed to be English ("en"). This element may occur zero to unlimited times in a software identification tag.		
Data Structure	**XML tag**	**Type**	**Definition**
	lang	XML character string. This is an optional tag attribute	The language the abstract is written in. Languages shall be specified as defined in IETF RFC 4646 - http://www.ietf.org/rfc/rfc4646.txt.
Example	<tag_creator_copyright lang="en">This tag may be used by used, stored, referenced and distributed by any software tool provider and or third party tag collection agency as long as the following elements are not modified: - entitlement_required_indicator - product_title - product_version - software_creator - software_licensor - software_id - tag_creator Extended information may also be added to the tag. </tag_creator_copyright>		

8.4.27 Tag version ('tag_version')

<table>
<tr><td>XML Tag</td><td colspan="3">tag_version</td></tr>
<tr><td>Type</td><td colspan="3">Complex Type</td></tr>
<tr><td>Definition</td><td colspan="3">This element provides a data element for tag creators or tag modifiers to provide tag version information. A properly defined software identification tag does not need to have a version specified by the tag creator since every software identification tag is unique. However as tags move through the software lifecycle, multiple tag modifiers may want to make changes to elements they are allowed to modify and/or to add extended elements to a software identification tag. In these cases, a version reference is required. There is a need to allow multiple entities to provide their own version information, meaning this element may be included multiple times within a single software identification tag. Each time a version element is provided all elements within the version element are required items to ensure uniqueness.

This element may occur zero to unlimited times in the software identification tag.</td></tr>
<tr><td rowspan="4">Data Structure</td><td>XML tag</td><td>Type</td><td>Definition</td></tr>
<tr><td>name</td><td>XML character string

One entry</td><td>This element provides the name of the entity defined in the tag. This name should be consistent between software products and software releases.</td></tr>
<tr><td>regid</td><td>regid type

One entry</td><td>Regid of the software creator (as specified in section 6.1.3.) If the entity is unknown, or is no longer in business, this value may be set to "unknown".</td></tr>
<tr><td>numeric_version</td><td>ProductVersion ComplexType - Complex type consisting of four elements with numeric values: “major”, “minor”, “build”, “review”

One entry</td><td>Numeric version identifier</td></tr>
<tr><td>Example</td><td colspan="3"><pre>
<tag_version>
  <name>My Example Corp</name>
  <regid>regid.1995-09.com.example</regid>
  <numeric_version>
    <major>1</major>
    <minor>0</minor>
    <build>0</build>
    <review>0</review>
  </numeric_version>
</tag_version>
</pre></td></tr>
</table>