Agnitio Advisors

Knowledge in Software Asset Management

Software Identification (SWID) Tags
A Strategic Solution

An Agnitio Advisors White Paper
by Paul 'Doc' Burnham

Agnitio Advisors
1484 Pollard Road, #135
Los Gatos, CA 95032
Phone: 408-871-0482
www.agnitioadvisors.com

Contents

Agnitio Advisors

Knowledge in Software Asset Management

Introduction

Organizations understand that software is an expensive and valuable asset that needs to be managed effectively. Even if an organization has some level of software asset management (SAM) practice in place, the use of software identification (SWID) tags offers a leap forward for SAM tools and processes, promising accurate, automated identification of installed software.

Many organizations believe that in order to benefit from SWID tags, they must wait for software publishers to provide the tags. In fact, end-user organizations can create and deploy their own SWID tags and reap the benefits now. This white paper explains how.

Benefits of Effective SAM Practices

Organizations with effective SAM practices are able to answer critical questions such as:

- What software entitlements do we own?

  Software entitlements are the rights to use software. Robust SAM practices require a comprehensive set of data about each software product owned or used by the organization, including software title, version, edition, language, what rights to use have been acquired, and any limitations on who and how the software may be used.

- What software is deployed/installed within the organization?

  Knowing what software is deployed or installed within an organization's environment requires effective and complete discovery procedures that can accurately identify software items that may be installed and in use on any device both inside and outside the organization's physical boundaries. Cloud services, virtualization and mobile devices continually stretch the number of systems that must be discovered and the complexities of finding the devices and software.

- What software is actually in use throughout our organization?

  Knowing what software is deployed and installed is only half the equation. It's also necessary to understand what software is actually being used, by whom, how and where.

The ability to accurately identify software that's installed and used provides organizations with the data needed to eliminate risks and reduce costs.

The ability to answer these questions provides organizations with the knowledge needed to reduce or eliminate various risks and costs both directly and indirectly associated with SAM. An obvious concern directly associated with SAM is license management, however, SAM impacts many IT and business processes including financial management, platform stability, productivity, security, business continuity and corporate/regulatory policies. Let's briefly review each of these areas.

- License management is about understanding what software is licensed, how it is licensed and how the software is used. For example: selecting and using the "right" license by metrics (by user, by processor, by site, etc) can provide cost savings through license optimization, flexibility in how and who can use the software, and reduce license management efforts. Organizations also benefit from reducing or eliminating both under-licensing and over-licensing of software. Proper licensing benefits include removing or lessening the risks related to unbudgeted software expenses and penalties for license non-compliance; as well as cost savings by not spending money on unneeded software.

"SAM impacts many IT and business processes including license management, financial management, platform stability, productivity, security, business continuity and corporate/regulatory policies."

- Financial management concerns the cost and control of software assets. This includes managing costs associated with purchasing and renewing licenses and software maintenance and support as well as accounting for those software assets. While these efforts can be quite challenging, they are necessary for organizations to demonstrate good financial stewardship.
- Platform stability refers to negative impacts on critical business applications or network stability through interoperability issues caused by the installation and use of non-approved or unauthorized applications. These unauthorized applications may not pose a security issue, but can still cause major disruption.
- Productivity relates to version or application incompatibility. Incompatibility issues such as file or format changes between versions can cause productivity challenges. While there are often workarounds for these problems, a lot of time and resources can be wasted in finding and implementing these workarounds.
- Security is about potential compromises to network or corporate data security. Protecting the integrity of the network and data means identifying real or potential threats and eliminating them. So whether the threat is an application not being at the correct patch level or some unauthorized application such as malware or peer-to-peer file sharing being installed, the ability to quickly and accurately identify and eliminate the threat is critical.

- Business continuity covers not only disaster recovery but also IT service management. Knowing which applications are installed on which machines is critical in a disaster recovery situation, but it is also critical to IT service management. The inability to quickly and accurately determine which applications are affected when a particular machine fails can be the difference between a minor system outage and a major system failure resulting in enormous costs.
- Corporate/regulatory policies apply to those requirements an organization has defined itself or that a regulatory agency has specified for the organization. These may include policies such as specifying which user roles or devices may have a specified software title installed to which regions may collect software usage data to taxation requirements on software titles installed ***in*** a particular region. In the EU, software usage data collection may be restricted by European Work Councils' rules, where European Work Councils exist as required by the European Works Council Directive.

Why Use Software Identification (SWID) Tags?

There are various methods used to identify software, each having their strengths and weaknesses. Because of the challenges related to accurately identifying software, some still consider it a 'dark art' (Hastings, 2010). These challenges have led to the development of software management and SAM tools that incorporate a hybrid of software identification methodologies to provide better and more accurate software identification. Regardless of the sophistication of the methods developed, no single tool will ever be able to accurately identify all software in use at any one time – the exercise is similar in some ways to archeology where people attempt to give an educated guess at what an animal may look like based on the remains that are unearthed from an archeological dig.

"Accurate identification of software allows organizations to support internal requirements and comply with regulatory requirements."

Proprietary methods are not needed if organizations – and the wider software publishing industry – adopt the ISO/IEC 19770-2:2009 software identification tag standard.

SWID tags are the key to more controlled and automated SAM practices, delivering the benefits described earlier at minimum cost and requiring minimal manual processing. Beyond this, accurate software identification contributes to achieving numerous other organizational objectives, for multiple teams across the organization. Accurate software identification helps:

- SAM teams know which applications are installed where, in order to validate license compliance
- IT security teams know if there are any installations of software with known security risks
- Desktop and server support teams know where particular software is installed in order to effect upgrades or problem resolutions
- Disaster recovery teams know which applications are installed on particular machines at particular locations, to facilitate disaster recovery exercises and operations
- IT service management teams know what software is on which servers, to understand what applications are affected during network connectivity or device failures
- Procurement teams know what licenses are being used, to facilitate annual renewals of maintenance and support contracts
- Internal audit teams know if any unauthorized applications are installed, in order to identify and eliminate risks they pose.

The SWID tag is an XML file containing authoritative identification and management information about a software product.

In addition to these teams' internal requirements, there are also various regulatory requirements dictating the ability to identify software. These include:

- Sarbanes-Oxley (SOX)
- Health Insurance Portability and Accountability Act (HIPAA)
- Electronic Records and Electronic Submissions CFR 21 part 11
- Financial Modernization Act of 1999 (GLBA)
- Federal Desktop Core Configuration (FDCC) and
- Presidential Executive Order 13103.

It is safe to declare that organizations need the ability to accurately identify software and should consider it a priority.

Overview of a SWID Tag

ISO/IEC 19770-2:2009 outlines and explains that the SWID tag is an XML file containing authoritative identification and management information about a software product. A SWID tag consists of data elements organized into three groups (Mandatory, Optional and

Extended) and placed in defined locations based on the operating system on which the software will be installed. These elements allow for accurate identification of software whether by automated means by tools with knowledge of SWID tag structures, or by tools that may not have the ability to process a SWID file directly, but can collect and store the SWID files for later processing.

Group	Comments
Mandatory	• 7 pre-defined elements • Required for a SWID tag to be considered valid or complete.
Optional	• 30 pre-defined elements • May or may not be provided • Allows tag creator additional opportunities to improve reliability of information for SAM practitioners and tool providers.
Extended	• No pre-defined elements • Tag creator may define unlimited elements • Allows tag creator additional opportunities to include values that have not been previously defined.

How to Use Software Identification (SWID) Tags

This section outlines how organizations can create and use their own SWID tags ("end-user SWID tags") to accomplish accurate software identification of the applications that are installed across their enterprise – whether the applications are commercial-off-the-shelf (COTS) or developed in-house – in a manner that provides strategic benefits to the organization as a whole.

Organizations can create and use their own SWID tags, whether the applications are commercial-off-the-shelf or developed in-house.

Defining Data Elements

Organizations have two options to consider before creating their end-user SWID tags. They can choose to use:

- Only the pre-defined mandatory and optional elements
- Both pre-defined mandatory and optional elements, together with their own defined extended elements.

If the organization decides to use extended elements, it is also required to create the XML schema to interpret those extended elements' data (refer to ISO/IEC 19770-2:2009, section 8.5). This XML schema (or XSD file) must be accessible to the organization's SAM tool, so it is commonly posted inside the organization's network. Using extended elements requires a limited amount of additional work for the organization, but ensures they can specify and later identify additional data elements for use in their SAM processes. The Appendix to this paper shows a worked example of how one organization defined its end-user SWID tags, including extended elements.

To start utilizing end-user SWID tags, the end-user organization need only modify their current processes slightly in order to add the steps defined in the sections *Creating End-user SWID Tags* and *Deploying End-user SWID Tags*.

Creating End-user SWID Tags

Implementing end-user SWID tags requires a minimal modification to current desktop management processes.

End-user organizations can create their SWID tags manually or using an automated process. Regardless of how a SWID tag is created, it provides the same ability to accurately identify software.

Manual Creation	Automated Creation
The manual method requires the tag creator to build the SWID tag XML file using any text editor. The tag creator must understand and validate the SWID tag's XML code.	The automated method can use either a tool or another automated process. The tool or automated process gathers the required data, and then creates the SWID tag. The automated process should also validate that the data entered conforms to the format in ISO/IEC 19770-2:2009 **Note: TagVault.org has a SWID tag creation tool available to members.**

Deploying End-user SWID Tags

Organizations can use end-user SWID tags whether they distribute their software via automated or manual means. The difference is in the process used to deploy the actual end-user SWID tag to the defined locations outlined by in ISO/IEC 19770-2:2009, section 6.1.4:

- Using automated distribution of software, the SWID tag is deployed as part of the “install package”. For example: for Windows installations, the “transform” function is used to add the SWID tag to the “install package” .msi file.
- Using manual distribution of software, the SWID tag is deployed manually to the defined locations.

Note that the defined locations for SWID tags depend on the operating system on which the software will be installed.

Using the Deployed End-user SWID Tags

Depending on the SAM tool used, the ability to take advantage of the data provided by the SWID tag will vary. Tools such as SmartTrack (from Aspera); IT Client Manager and IT Asset Manager (from CA); K2 KeyAuditor and KeyServer (from Sassafras Software); Altiris Inventory Solution (from Symantec); and Software Management Suite 2010 (from SoftwareManagement.org) have incorporated the ability to use SWID tags and the data they provide.

Other SAM tools, even though they have not incorporated the automatic use of SWID tags, can also use the data provided by SWID tags. For example, simply locating SWID tag files (identified by their .swidtag file extension) provides the organization with the knowledge of whether a SWID file exists or not. Additionally, the SWID tag’s name can itself provide a certain level of software identification. And of course SWID tag files can be looked at manually to gain even further information.

In early 2010, The ITAM Review and TagVault.org jointly sponsored the “Best Use of Tags 2010” contest to highlight the abilities of SAM tools to utilize SWID tags. See www.tagvault.org/node/116

Conclusion

The ability to accurately identify software is not only essential, but foundational to achieve effective software asset management.

After working in the SAM environment, even for a short time, it quickly becomes apparent that the ability the accurately identify software is not only essential, but foundational to achieve effective SAM.

The importance of software identification is evident when assessing SAM practices using either the SAM Optimization Model (http://www.microsoft.com/sam/en/us/optmodel.aspx) or other assessment models based on ISO/IEC 19770-1:2006. For example, when using the SAM Optimization Model to assess SAM practices, software identification is not only critical for the Hardware and Software Inventory competency, but vital for seven other key competencies.

As the understanding of SWID tags increases, more organizations will begin to use them to provide a solution that complements their current SAM practices by accurately identifying software, while at the same time enhancing organizational ability to overcome license management, platform stability, productivity, security, business continuity issues and organizational/regulatory policy requirements.

Organizations can start reaping the benefits now, without significant investment, by implementing end-user SWID tags. End-user developed SWID tags can be utilized even while these organizations require certified tags from their software publishers. The market has numerous examples of organizations that are already starting to require ISO/IEC 19770-2:2009 SWID tags or TagVault.org-certified tags from their publishers (see http://tagvault.org/DMMI_Product_Manager_sees_value_in_19770_standards).

Due to the fact that SWID tags provide explicit details about the software installed on a device it finally provides a way for end-user organizations to balance the audit requirements imposed by software publishers with a requirement that those publishers provide explicit details for the organization to discover and authoritative identify software themselves (http://tagvault.org/RFP_and_Contract_Language).

Recommended Reading

For more details about the uses and structure of SWID tags, we recommend reading these white papers, available at www.TagVault.org/white-papers:

- *Using ISO/IEC 19770-2 Software Identification Tags to Enhance Software Asset Management*, by Paul 'Doc' Burnham for Agnitio Advisors, 2010
- *Why Software License Management is So Difficult — and How To Simplify It*, by Howard Hastings for CA, 2010 (also available for download at www.ca.com/files/whitepapers/software-license-management-wp.pdf)
- *The Future of Software Asset Management*, by Steve Klos for TagVault.org, 2010

Also see:

The TagVault.org website, www.TagVault.org

The SAM Guides website, www.SAMguides.com

The ISO/IEC 19770-2:2009 standard is available from ISO at
http://www.iso.org/iso/rss.xml?csnumber=53670&rss=detail
and from ANSI at
http://webstore.ansi.org/RecordDetail.aspx?sku=ISO%2fIEC+19770-2%3a2009
or from your national standards organization.

The XSD that defines the structure of SWID tags can be found at
http://standards.iso.org/iso/19770/-2/2009/schema.xsd

About the Author

Paul 'Doc' Burnham has more than 20 years IT experience with an extensive background in software asset management (SAM), software compliance, IT asset procurement, contract and vendor management, and financial analysis. Doc is recognized for his expertise in both the mechanics and the implementation of SAM. His activities within the SAM ecosystem have included development work on ISO/IEC 19770 series, developing and presenting ITAM- and SAM-related training, authoring white papers, and working jointly with other SAM practitioners, tool vendors and software publishers. Doc has also been recognized as an IAITAM fellow.

Agnitio Advisors

Knowledge in Software Asset Management

About Agnitio Advisors

Agnitio Advisors provides consulting services to assess, develop and optimize organization IT practices. Projects are structured around industry standards including ISO/IEC 19770-1, the Microsoft SAM Optimization Model and ITIL.

Agnitio Advisors applies years of experience from hundreds of projects in organizations of all sizes and market segments to ensure the correct projects are implemented at the right stage of organizational maturity to maximize customer benefits. Projects are based on a pragmatic approach to cost savings, cost avoidance and process efficiency. Managing software effectively provides significant benefits beyond the traditional IT Asset Management (ITAM) focus. Agnitio Advisors can help organizations establish more effective and measurable security policies, improve device stability and decrease help desk costs, among other things.

Agnitio Advisors' consultants have real world experience in auditing, implementation and tool evaluation as well as a high degree of focus on policies, processes and workflows for effective IT practices. Through these experiences, Agnitio Advisors has been recognized by many industry organizations as highly valued partners. A few highlights of the recognitions received by Agnitio Advisors employees include:

- *Microsoft Certified Partner with a competency in Software Asset Management*
- *Recognized as Fellows of the International Association of IT Asset Management (IAITAM)*
- *Regular judges for the SIIA CODiE awards in the Best Asset Management category*
- *Certified trainers in multiple industry based SAM training courses*
- *Developed training materials for Microsoft SAM partners focused on baseline, assessment and deployment planning projects*
- *Convener for the ISO/IEC 19770-2 standard for Software Identification Tags*
- *Selected to help in the initial formation of the IEEE-ISTO non-profit organization TagVault.org*

Agnitio Advisors consultants are brought into a project for their knowledge, practical experience and value-based approach to assisting customers. Agnitio Advisors remains involved in projects due to their ability to work effectively within a customer's requirements and resource limitations.

For more information about Agnitio Advisors, contact us or visit us on the web at http://www.agnitioadvisors.com.

References

- Preview of the ISO/IEC 19770-2 software identification tag standard (includes the first 3 sections of the document), http://webstore.iec.ch/preview/info_isoiec19770-2%7Bed1.0%7Den.pdf

  ISO and ANSI webstores to purchase the standard, http://www.iso.org/iso/rss.xml?csnumber=53670&rss=detail http://webstore.ansi.org/RecordDetail.aspx?sku=ISO%2fIEC+19770-2%3a2009
- Schema for SWID XML structure, http://standards.iso.org/iso/19770/-2/2009/schema.xsd
- Hastings, Howard, Why Software License Management is So Difficult – *and How to Simply It*, 2010, www.ca.com/files/whitepapers/software-license-management-wp.pdf
- TagVault.org suggestions to add certified tags into purchased software, http://tagvault.org/RFP_and_Contract_Language
- GSA efforts to integrate SWID tags in the purchasing process, http://www.gsa.gov/portal/content/103237
- Details on certification documentation defined by the GSA Working Group, www.tagvault.org/GSA_Working_Group_Certification_Document
- TagVault.org Whitepapers, www.TagVault.org/white-papers
- SAMGuides reference website, www.SAMguides.com
- Best Use of SWID tags contest information www.tagvault.org/node/116

Agnitio Advisors

Knowledge in Software Asset Management

Appendix

Case Study – Using Software Identification (SWID) Tags

This case study shows how one organization defined SWID tags for its environment to enhance software identification. The implementation and use of end-user SWID tags was highlighted to senior management as the key step that enabled their SAM program to better manage their software assets and document over $ 1 million in value (cost savings/avoidance in funds and work hours) within six months of implementation.

The implementation and use of end-user SWID tags was the key step that enabled their SAM program to document over $ 1 million in cost savings/avoidance in funds and work hours within six months of implementation.

Some facts about this organization's approach to SAM and SWID tag creation:

- They did not have a SAM tool that takes full advantage of the SWID tags. Instead, they modified the tool slightly and modified some of their existing processes.
- They created the SWID tags manually.
- They deployed software both manually and automatically within their environment of over 5000 desktops and 300 servers.

For anonymity reasons, the name and related data for the case study organization has been replaced with Agnitio Advisors' name and data.

Defining Data Elements

Based on ISO/IEC 19770-2:2009, this organization incorporated 22 elements in their end-user SWID tags:

- All seven pre-defined mandatory elements
- Eight of the 30 pre-defined optional elements
- Seven user-defined extended elements.

The following table provides information about the elements used, including the objective for using the element, its value, how it was created and any additional insights about the element or the organization's use of it.

The organization adopted this format as its standard, so that each of their end-user SWID tags contained the same elements.

The organization will be transitioning to the TagVault.org certified SWID tagging requirements for all element values. This will ensure that as commercial software is deployed with SWID tags, that the reporting and management processes will be consistent.

Note that extended elements may be added to commercial software identification tags as wel

Data Element	Purpose	Value and Example Data Format	Comments
Mandatory Elements			
Entitlements required indicator (entitlement_required_ indicator)	Identifies whether this application needs to match up with an entitlement for a reconciliation to be considered successful	Specifies if an entitlement is required to reconcile this application. See ISO/IEC 19770-2:2009 for guidance. Example data format: XX = entitlement status (true = entitlement required; false = no entitlement required)	
Product title (product_title)	Identifies the application	Accurately identifies the application. See ISO/IEC 19770-2:2009 for guidance.	
Product version (product_version)	Accurately identifies the application version	Identifies the application version using two values – numeric version and version string. See ISO/IEC 19770-2:2009 for guidance.	
Software creator identity (software_creator)	Identifies the creator of the software	Identifies the application creator. See ISO/IEC 19770-2:2009 for guidance.	
Software licensor identity (software_licensor)	Identifies the licensor who owns the software copyright	Identifies the application licensor. See ISO/IEC 19770-2:2009 for guidance.	
Software unique identifier (software_id)	Identifies the specific version of a specific application	Accurately identifies the application. See ISO/IEC 19770-2:2009 for guidance.	

Data Element	Purpose	Value and Example Data Format	Comments
Tag creator identity (tag_creator)	Identifies the tag creator	Identifies the tag creator. See ISO/IEC 19770-2:2009 for guidance. Example data format: “regid.2007-01.com.agnitioadvisors,BU-A” where: • regid = registered id • 2007-01 = the year and first month the domain was registered (yyyy-mm) • com = the upper level domain • AgnitioAdvisors = the domain name • BU-A = the sub-unit creating the SWID tag • xxx = the name of the business unit (BU) that created the SWID tag Note that everything after the comma ‘,’ is optional and only required if a software title is specific to a particular business unite.	
Optional Elements			
Abstract (abstract_application_ info)	Provides brief information on application	Used to provide information related to the application, such as an overview of what the application is or does; special instruction on restrictions, requirements, etc. Example data format: • "application is the company standard for office installations"	

Data Element	Purpose	Value and Example Data Format	Comments
Component association (component_of)	Typically used to identify patches and other add-on type software installations that can be added to a device after a product installation. Example: Microsoft Save as PDF or XPS is an add-on component of Microsoft Office 2007	Used to provide child / parent information related to the application, for example that the application is a component of software X. Example data format: • unique_id = identifier of parent application • Tag_creator_regid = regid of who provided conformation of component relationship	1. This indicates that this package is part of a bundle or is a component of another software product i.e. application x is part of bundle Core-Image-1.
Components list (complex_of)	Provides information where an application is a complex of (or parent of) other packages. This element will typically indicate if the identified software is a bundle of other products, or a suite product.	Used to provide child/parent information related to the application, for example that this application is the parent of package a, package b, package c. Example data format: • Unique_id_1 = identifier of child application 1 • Unique_id_2 = identifier of child application 2 • Tag_creator_regid = regid of who provided conformation of child/ parent relationship	1. The number of child applications will various, so add lines as required 2. This element would only be present if the software was a parent of other software package
License and channel information (license_linkage)	Provides information on application license linkage	Defines the channel and type of license (i.e. OEM vs. Volume) that can affect the user's rights. See ISO/IEC *19770-2:2009 for guidance.* Example data format: • activation_status id=e8_9sub1 = X • channel_type id="e8_4_9sub2 = X • customer_type id=e8_4_9sub3 = X	1. When certified software identification tags are used, they will use a standard set of values for this element. Where possible, end-users should also use these standard values, which can be found in the TagVault.org GSA certification

Data Element	Purpose	Value and Example Data Format	Comments
			document (refer to www.tagvault.org/GSA_Working_Group_Certification_Document for further details). 2. This element may not be present when SWID tag is for end-user defined bundles (i.e. Core-Image-1).
Product category (product_category)	Identifies and categorizes the type of software this SWID tag refers to	Specifies the UNSPSC category that applies to this software product: Example data format: • commodity_title=Enterprise resource planning ERP software • code=43231602</code>	1. For codes, refer to www.unspsc.org. Most codes for software will be found in the 4323xxxx section of the code. This code should be the same code used within the procurement system
Release identifier (release_id)	Identifies the package release	Defines each particular release of the install package, this is helpful when looking into root cause issues. Example data format: • AA-ENG-Microsoft-OfficeProPlus-2007wSP2-ENT-R1 where: • XX = entity the install package is for (AA = Agnitio Advisors, MSFT = Microsoft, etc) • a good method is to use the stock symbol if available • XXX = Language (ENG = English, ESP = Spanish, etc) • Publisher • Application name	1. When the package is modified, its release variation (e.g. R1 to R2) will also change, which in turn requires a new SWIDtag to reflect the change.

Data Element	Purpose	Value and Example Data Format	Comments
		• Version • XXX = entity (ENT = enterprise, BU-X = sub-unit X, etc) • Release variation (R1 = Release 1, R2 = Release 2, etc)	
Serial number (serial_number)	Identifies if a serial number (or key code) is required for the application, if so where that serial number can be found.	Identifies the serial number used in the application, this is especially helpful in the case where multiple serial numbers are available based on the licenses. • Example data format: Microsoft_select_7654321_Office-2007-suites • XX = none or reference to key code location	It is not recommended to include the actual serial number/key code
Upgrade for (upgrade_for)	Identifies this application as an upgrade for a specific application	Provides information on application and license progression. Example data format: • Upgrade_id = identifier of upgraded application • Tag_creator_regid = regid of who provided conformation of upgrade path • Upgrade_description = details of the upgrade that SAM/ITAM team should be aware of	This element would only be present if software was an upgrade for another application.
Extended Elements The use of extended elements required the end-user organization to create the XML schema (XSD file) to interpret the data (refer to section 8.5 of ISO/IEC 19770-2 for details about the requirement for, and structure of, the XML schema). Information included in this section was identified as being required by the organization creating and distributing the SWID tags. Your organization should determine what data should be tracked – these are just examples from one organization.			

Data Element	Purpose	Value and Example Data Format	Comments
Tag creator email (tag_creator_email)	Identifies the tag creators email - this may be helpful to organizations that do not have an effective package management infrastructure that would identify the creator of a SWID tag.	Provides a contact email for the tag creator. Example data format: • esam@agnitioadvisors.com • XX = the email of the enterprise SAM team	A good rule to follow is to create a common email address for the ITAM or SAM team.
Install package identifier (install_package_id)	Identifies the complete SWID tag name for the install package.	Identifies what the complete SWID tag name will be for this particular install package. This is particularly useful when manually creating the SWID tags. Example data format: • "regid.2007-01.com.agnitioadvisors,BU-A_AA-ENG-Microsoft-OfficeProPlus-2007wSP2-ENT-R1.swidtag" • This is made up of the following three items: • tag creator id (tag_creator_id) - regid • release identifier (release_id) • file extension (.swidtag)	
Application license type (application_licensetype)	Identifies the type of license the application is licensed under and used in concert with the Optional element - License and channel information. .	Identifies the license type by which the application is licensed. Example data format: • "perpetual-volume" • XX = license type (time-based (lease, subscription, etc); perpetual (OEM, Retail, Volume, Academic, etc); or None (when related to end-user defined bundles)	This element was used because the organization decided to build different software packages based on license type. This provided them additional details when accounting for the deployed licenses (i.e. OEM licenses have different usage rights them Volume licenses).

Data Element	Purpose	Value and Example Data Format	Comments
Application license by (application_licenseby)	Identifies the metric the application is licensed by	Identifies the license metric by which the application is licensed. Example data format: • XX = license by metric (none, device; processor; core; user; named user; etc)	This element was used because the organization decided to build different software packages based on license by metric. This provided them additional details when accounting for the deployed licenses (i.e. identifying device vs. processor licenses). This did require the organization to remove and replace the software (or at least the SWID files) when they changed license by metrics (i.e. switching Microsoft SQL server from server to processor licensing).
CAL data (CAL_data)	Identifies if a specific Client Access License (CAL) is required, and if so, the name of the CAL	Identifies which CAL, if any, is required for this application. Example data format: • XX = none or name of CAL (Microsoft_sql_user, Microsoft_exchange-enterprise_user, Microsoft_windows_device, IBM_DB2_user, etc)	
Application control reference (application_control_ref)	Identifies if the application is controlled	Identifies controls, if any, related to this application. Used to determine if controlled applications are deployed outside the controlled area. Example data format: • none • XX = none or identify control metric (country, business unit,	

Data Element	Purpose	Value and Example Data Format	Comments
		region, etc)	
Application usage reference (application_usage_ref)	Identifies how to validate usage of this application	Identifies how to determine 'usage' of the application Example data format: • access_installed • XX = none or identify usage metric (access, installed, etc – this may list multiple metrics to measure)	

Example 1 End-user SWID Tag

The following example is the end-user SWID tag created for Microsoft Office Professional Plus 2007 with SP2, based on the format identified above. The highlighted data is the data the end-user was required to complete.

Note: Since extended elements were used, there was a requirement to create and publish a user-defined XML schema.

```
<?xml version="1.0" encoding="UTF-8"?>
<swid:software_identification_tag
        xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:swid="http://standards.iso.org/iso/19770/-
2/2008/schema.xsd"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://standards.iso.org/iso/19770/-2/2009/schema.xsd
software_identification_tag.xsd ">
        <!-- Mandatory elements -->
        <swid:entitlement_required_indicator id="e8_3_1">true</swid:entitlement_required_indicator>
        <swid:product_title id="e8_3_2">Microsoft-OfficeProPlus-2007wSP2</swid:product_title>
        <swid:product_version id="e8_3_3">
          <swid:name id="e8_3_3sub1">12.0</swid:name>
          <swid:numeric id="e8_3_3sub2">
                  <swid:major id="e8_3_3sub2sub1">12</swid:major>
                  <swid:minor id="e8_3_3sub2sub2">0</swid:minor>
                  <swid:build id="e8_3_3sub2sub3">0</swid:build>
                  <swid:review id="e8_3_3sub2sub4">0</swid:review>
          </swid:numeric>
        </swid:product_version>
        <swid:software_creator id="e8_3_4">
```

```
          <swid:name>Microsoft Corporation</swid:name>
          <swid:regid>regid.1991-06.com.microsoft</swid:regid>
        </swid:software_creator>
        <swid:software_licensor id="e8_3_5">
          <swid:name>Microsoft Corporation</swid:name>
          <swid:regid>regid.1991-06.com.microsoft</swid:regid>
        </swid:software_licensor>
        <swid:software_id id="e8_3_6">
          <swid:unique_id>Office-Pro-Plus-2007(wSP2)</swid:unique_id>
          <swid:tag_creator_regid> regid.2007-01.com.agnitioadvisors,BU-A</swid:tag_creator_regid>
        </swid:software_id>
        <swid:tag_creator id="e8_3_7">
          <swid:name>Agnitio_Advisors</swid:name>
          <swid:regid>regid.2007-01.com.agnitioadvisors,BU-A</swid:regid>
        </swid:tag_creator>
        <!-- Optional elements -->
<swid:abstract_application_info id="e8_4_1">application is the company standard for office
installations</swid:abstract_application_info>
        <swid:license_linkage_id ="e8_4_9">
          <swid:activation_status id="e8_4_9sub1">Licensed</swid:activation_status>
          <swid:channel_type id="e8_4_9sub2">Volume</swid:channel_type>
          <swid:customer_type id="e8_4_9sub3">Corporate</swid:customer_type>
        </swid:license_linkage>
        <swid:product_category-id="e8_4_12">
          <swid:commodity_title>Office-Suite-Software</swid: commodity_title>
          <swid:code>43231513</swid:code>
        </swid:product_category
<swid:release_id id="e8_4_16">AA-ENG-OfficeProPlus-2007wSP2-ENT-R1</swid:release_id>
        <swid:serial_number id="e8_4_20">Microsoft_select_7654321_office-2007-
suites</swid:serial_number>
        <!-- Extended elements -->
        <swid:extended_information>
          <exti:xmlns:exti="http://www.agnitioadvisors.com/eSAM_swidtag_ExtendedInfo.xsd"
            xsi:schemaLocation="http://www.agnitioadvisors.com/eSAM_swidtag_ExtendedInfo.xsd
AgnitoAdvisors_tag_schema.xsd">
           <swid:tag_creator_email id="ext_1_1">esam@agnitioadvisiors.com</swid:tag_creator_email>
          <swid:install_package_id id="ext_1_2"> regid.2007-01.com.agnitioadvisors,BU-A_AA-ENG-Microsoft-
Office-Pro-Plus-2007(wSP2)-ENT-R1.swidtag</swid:install_package_id>
```

```
<swid:application_licensetype id="ext_1_5">perpetual-volume</swid:application_licensetype>
<swid:application_licenseby id="ext_1_6">device</swid:application_licenseby>
<swid:cal_data id="ext_1_7">none</swid:cal_data>
<swid:application_control_ref id="ext_1_8">none</swid:application_control_ref>
<swid:application_usage_ref id="ext_1_9">access_installed</swid:application_usage_ref>
</swid:extended_information>
</swid:software_identification_tag>
```

Example 2 End-user SWID Tag

The following example is the end-user SWID tag created for Core-Image-1-Sep10, based on the format identified above. The highlighted data is the data the end-user was required to complete.

Notes:

Since extended elements were used, there was a requirement to create and publish a user-defined XML schema

Not all the applications were listed under the “complex_of” element.

```
<?xml version="1.0" encoding="UTF-8"?>
<swid:software_identification_tag
        xmlns:ds="http://www.w3.org/2000/09/xmldsig#" xmlns:swid="http://standards.iso.org/iso/19770/-
2/2008/schema.xsd"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://standards.iso.org/iso/19770/-2/2009/schema.xsd software_identification_tag.xsd
">
        <!-- Mandatory elements -->
        <swid:entitlement_required_indicator id="e8_3_1">true</swid:entitlement_required_indicator>
        <swid:product_title id="e8_3_2">AA-Core-Image-1-Sep10</swid:product_title>
        <swid:product_version id="e8_3_3">
          <swid:name id="e8_3_3sub1">1.0</swid:name>
          <swid:numeric id="e8_3_3sub2">
                  <swid:major id="e8_3_3sub2sub1">0</swid:major>
                  <swid:minor id="e8_3_3sub2sub2">0</swid:minor>
                  <swid:build id="e8_3_3sub2sub3">0</swid:build>
                  <swid:review id="e8_3_3sub2sub4">0</swid:review>
          </swid:numeric>
        </swid:product_version>
        <swid:software_creator id="e8_3_4">
          <swid:name> Agnitio_Advisors </swid:name>
          <swid:regid> regid.2007-01.com.agnitioadvisors,BU-A </swid:regid>
        </swid:software_creator>
```

```
         <swid:software_licensor id="e8_3_5">
           <swid:name> Agnitio_Advisors </swid:name>
           <swid:regid> regid.2007-01.com.agnitioadvisors,BU-A </swid:regid>
         </swid:software_licensor>
         <swid:software_id id="e8_3_6">
           <swid:unique_id>Core-Image-1-Sep10</swid:unique_id>
           <swid:tag_creator_regid> regid.2007-01.com.agnitioadvisors,BU-A</swid:tag_creator_regid>
         </swid:software_id>
         <swid:tag_creator id="e8_3_7">
           <swid:name>Agnitio_Advisors</swid:name>
           <swid:regid>regid.2007-01.com.agnitioadvisors,BU-A</swid:regid>
         </swid:tag_creator>
         <!-- Optional elements -->
<swid:abstract_application_info id="e8_4_1">this image contains the core applications used on all pc and laptops
within Agnitio Advisors environment</swid:abstract_application_info>
         <swid:complex_of id ="e8_4_3">
           <swid:unique_id_1>AA-ENG-OfficeProPlus-2007wSP2-ENT-R1</swid:unique_id_1>
           <swid:unique_id_2>AA-ENG-AcrobatReader-9.3.4-ENT-R3</swid:unique_id_2>
           <swid:unique_id_3 >AA-ENG-WinZipPro-14.3-ENT-R2</swid:unique_id_3>
           <swid:unique_id_4>AA-ENG-Quicktime-7-ENT-R4</swid:unique_id_4>
           <swid:unique_id_5>AA-ENG-FlashPlayer-10-ENT-R3</swid:unique_id_5>
           <swid:tag_creator_regid id>regid.2007-01.com.agnitioadvisors,BU-A</swid:tag_creator_regid>
         </swid:complex_of>
         <swid:product_category-id=”e8_4_12”>
           <swid:commodity_title>Software</swid: commodity_title>
           <swid:code>43230000</swid:code>
         </swid:product_category
<swid:release_id id="e8_4_16">AA-Core-Image-1-Sep10-R1</swid:release_id>
         <swid:serial_number id="e8_4_20">none</swid:serial_number>
         <!-- Extended elements -->
         <swid:extended_information>
           <exti:xmlns:exti="http://www.agnitioadvisors.com/eSAM_swidtag_ExtendedInfo.xsd"
             xsi:schemaLocation="http://www.agnitioadvisors.com/eSAM_swidtag_ExtendedInfo.xsd
AgnitoAdvisors_tag_schema.xsd">
           <swid:tag_creator_email id="ext_1_1">esam@agnitioadvisiors.com</swid:tag_creator_email>
           <swid:install_package_id id="ext_1_2"> regid.2007-01.com.agnitioadvisors,BU-A_AA-Core-Image-1-Sep10-
R1.swidtag</swid:install_package_id>
<swid:application_licensetype id="ext_1_5">none</swid:application_licensetype>
<swid:application_licenseby id="ext_1_6">none</swid:application_licenseby>
<swid:cal_data id=”ext_1_7”>none</swid:cal_data>
<swid:application_control_ref id="ext_1_8">none</swid:application_control_ref>
```

```
<swid:application_usage_ref id="ext_1_9">none</swid:application_usage_ref>
</swid:extended_information>
</swid:software_identification_tag>
```