



About The Express Software Identification Database (ESID)®

The Express Software Identification Database (ESID) is a comprehensive catalog of commercial and free PC software applications that run on Windows desktops and servers. A key component of Express Metrix's own IT asset management solutions, the ESID provides a means for comprehensively and accurately identifying the wide range of software installed and used across corporate networks. The ESID and its associated recognition algorithms are also licensed for use within other products that require precise application identification to perform their functions. Companies that OEM the ESID include IBM, BMC, Numara Software, New Boundary, LANDesk, Adlon, and more.

The Software Identification Challenge

Any software product that needs to access, interact with, or report on installed desktop applications requires a method of 1) discovering and 2) recognizing those applications. Without strong application discovery and recognition capabilities, software data cannot be properly rationalized or presented in an accurate, unified, or truly meaningful manner.

Ascertaining the presence of applications on a PC is a fairly straightforward process from a technology standpoint. However, there exists no common, standardized method for correlating "discoverable" application data indicating the presence of an installed program with its actual software title, version, manufacturer, or other important application data. In fact, most products that claim to discover and identify applications in an automated fashion rarely deliver information reliable and accurate enough to calculate the correct number of installed titles. Most products rely on one of two methodologies: examining file header and analyzing registry entries. Due to the inherent lack of uniformity within the computing environment, the use of any one of these methods in isolation is inaccurate and/or incomplete.

File Header Identification

File header analysis is one commonly used technique for identifying installed applications. The advantage of this approach is that it's directly tied to the application executable; the presence of an application is determined by the presence of its executable(s). However, one major issue with file header analysis is the level of reliability of the information the file header contains – many software vendors don't provide complete information or don't update file headers on a regular basis, leading to inaccurate, inconsistent, or even missing application data. For example, the file header for Google's Chrome browser, when read on the Windows platform, reveals its name and copyright, but no version information at all. In addition, Adobe products are notorious for being named inconsistently, making it difficult to evaluate data in a consolidated, unified manner.

But even for applications that do include reliable information, a much more significant problem often exists. Applications frequently consist of numerous (sometimes hundreds or thousands) executable files, and in many cases, multiple applications from a single vendor may share one or more identical executables. Because file headers contain no information revealing the relationship between executables, there's no way of knowing with which application they correspond. The net results of basing application recognition on this information are: a) artificially inflated application counts that may lead to the wrong conclusion that too few licenses have been purchased than are actually required, and/or b) uncertainty as to which application a shared executable represents.

Registry Analysis

Many technologies rely on information contained in the Windows registry to determine what software is installed on a PC. However, there are some downsides associated with this methodology. First, programs installed using means other than the Windows installer are often not detected; and even when they are, they may lack critical version information. More importantly, registry information (such as that shown in the Add/Remove Programs control panel) is based on how a product is packaged from an installation perspective, not how the components of the product need to be evaluated from a licensing standpoint. In other words, each entry in Add/Remove does not necessarily indicate a separate application for which a license is required.

Software Catalogs

An alternative to the identification methods described above is the use of a software recognition database. Such "software catalogs" are generally built by teams who understand how to turn raw data into normalized information representing how software is actually licensed. Software catalogs are therefore usually more useful than the other methods for generating accurate license counts and evaluating one's license position. With this approach, special algorithms are used to allow discovered executable file and other application information to be compared with application data residing in the database. The end result is a list of normalized software titles, manufacturers, versions, and other information indicated by the discovered data. Further, software catalogs often contain other "non-discoverable" information originating outside the desktop environment—such as software categories, SKU information, and just about any other relevant data point—that can be associated with the identified application titles.

Assuming the algorithms are properly constructed, the success of the outcome depends on the accuracy and comprehensiveness of the software database. That is where the ESID comes in.

The Express Software Identification Database (ESID)

The ESID eliminates the need to rely solely on inaccurate or incomplete file headers and/or registry entries to identify applications within the desktop and server environment. The database contains an extensive collection of application titles and their associated details such as version, manufacturer, software category, executable file names, file sizes, and more. Application information (such as that contained within the Windows registry and/or file headers) collected by an agent can then be evaluated against the catalog data to correctly determine installed application titles and versions—a prerequisite for using the information in a meaningful way.

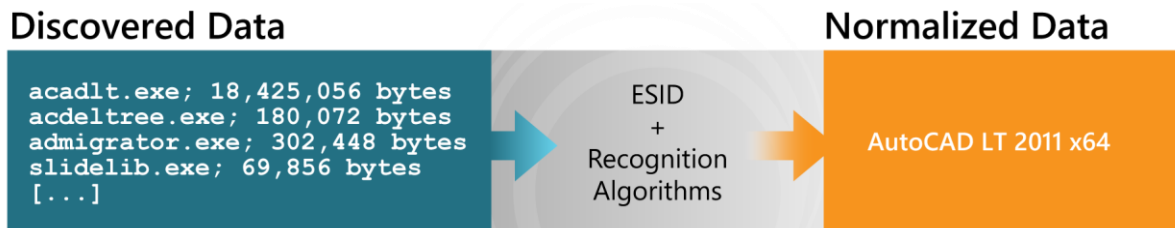
For most applications, this "executable signature" level of analysis is sufficient. However, in other situations, file signatures are not enough to reveal information that's critical for making licensing decisions. For example, executable name and size alone aren't enough to determine whether a copy of Microsoft Access 2007 was

installed as a standalone product, as part of the Microsoft Office 2007 Professional edition, or as part of the Enterprise edition. In such cases, further analysis using Globally Unique Identifiers (GUIDs) is then applied to complete the identification process.

Once applications and suites have been properly identified using the methods described above, the results can be analyzed and grouped in such a way as to facilitate a variety of software asset management tasks. Without such rationalization, capabilities such as the following would be virtually impossible:

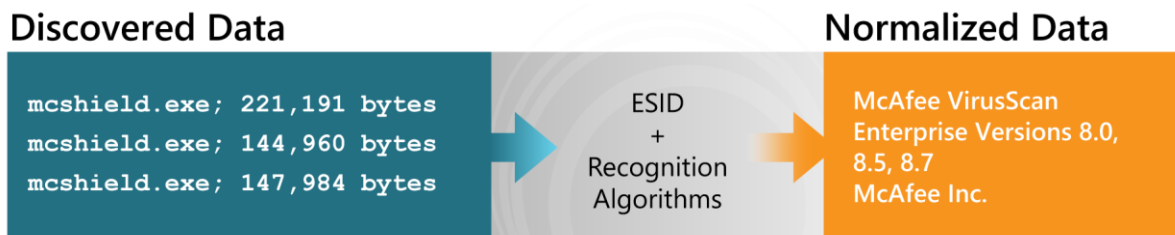
Group and correlate all executable files related to a single application

Most applications consist of more than one executable. Without proper recognition and rationalization, multiple related files may be incorrectly interpreted as independent applications or, in some situations, not identified at all. Such errors can only be ascertained and corrected manually, an overwhelmingly tedious and time-consuming process. The ESID contains executable information that enables your product to group all discovered executables that comprise a single application, so that data can be evaluated from a licensing perspective.



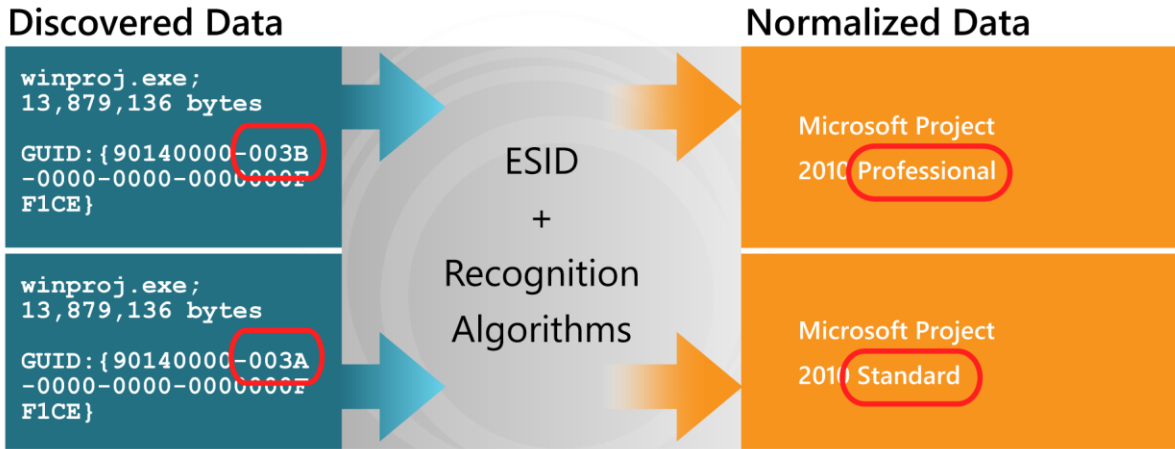
Normalize variations in application titles, versions, and manufacturer

Traditional methods of software recognition depend on software publishers to provide consistent information across all products and all product versions. Unfortunately, it's rarely the case that this information is consistent from release to release (let alone across an entire product line), even among the most reputable manufacturers. The ESID normalizes all the variations of any given software title, version, or publisher into consistent, familiar nomenclature. For example, software license agreements often allow for simultaneous versions or multiple copies of an application to be installed. Without a way of normalizing different file headers, discovered executables can't be grouped under a common application title. This can present a significant challenge from a license management perspective because each version or installation may be mistakenly viewed as a separate application requiring its own license. The ESID relies on executable file information that enables all versions of a given application to be consolidated under a single application title and manufacturer, while preserving the versioning information.



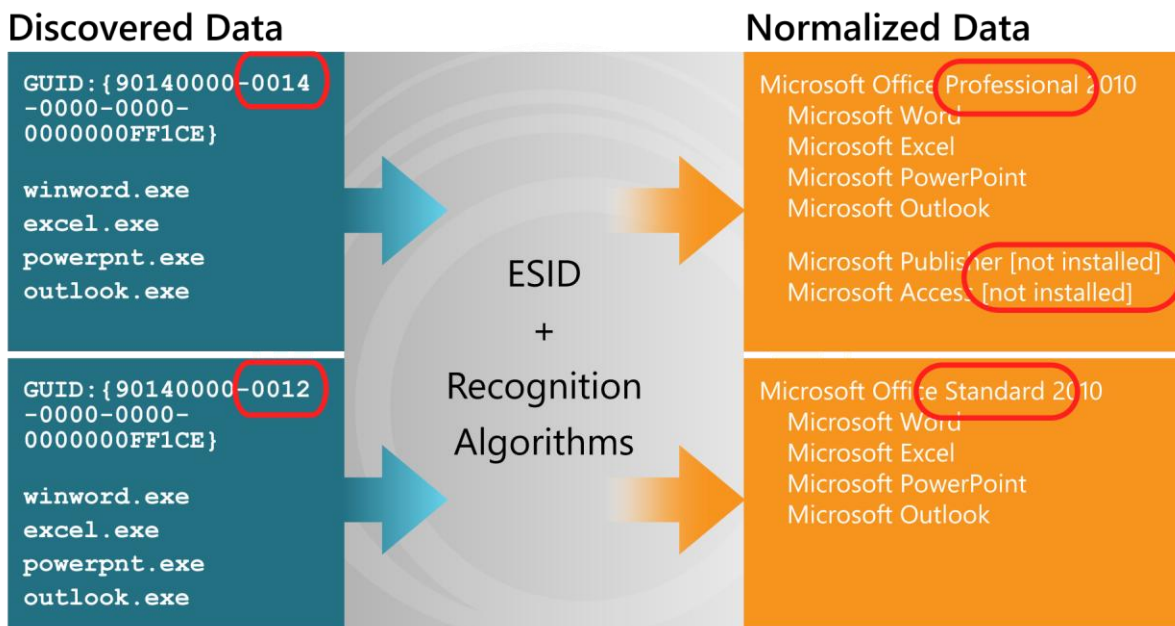
Differentiate between applications that share a common executable

Identification methods that rely solely on executable file information generally lack the detail necessary to distinguish between different editions of applications that share a common executable. This is the case, for example, with Microsoft Project Standard and Microsoft Project Professional—a critical distinction to make from a license and cost standpoint. The ESID also contains registry-based information (in the form of GUIDs) which enable proper identification of such applications.



Correctly identify suites and their installed components

Methods that analyze the registry will identify the presence of a suite but generally won't reveal which components of the suite are actually installed. This makes it extremely difficult, for example, to collect software usage data on individual programs in order to determine whether they are actually being utilized. Other techniques that rely on executable file information alone may recognize the individual suite components, but cannot provide the information necessary to identify the exact suite configuration. The ESID uses GUIDs in conjunction with executable file information, making such analysis possible.



The capabilities noted above are nothing short of critical when it comes to virtually every license management function. Because the ESID was designed to provide application identification utilizing multiple methods—and has specialists dedicated to researching and validating the accuracy of the data—one can appreciate why so many world-class software vendors partner with Express Metrix to bring superior application recognition to their own products.