Scalable Software Protection and Flexible Licensing

CodeMeter* and *CodeMeterAct

WIBU SYSTEMS

Version 1.0

Copyright

Ruepurrer Strasse 52-54 | 76137 Karlsruhe, Germany

Tel.: +49-721-93172-0 | Fax: +49-721-93172-22

E-Mail: info@wibu.de

Company	WIBU-SYSTEMS AG
Street Address	Rueppurrer Strasse 52-54
Postal Code, Place	76137 Karlsruhe
Country	Germany
Author	Rüdiger Kügler
Fon	+49 721 93172 - 0
Version	1.1

Table of Contents

1 Introduction

Software manages our life. You find software today not only on office PCs and in children's rooms but also embedded in every day machines – starting with the simple coffee maker, where a programmable controller interprets the bar code of an inserted pad, and then controls water quantity and temperature, up to knitting machines big enough to fill a warehouse. More often than not, the know-how is included in the controlling software.

However, the ubiquitous nature of software also has its down side. Software is easily digitally copied. The resulting pirated copy has the same quality as the original software. In the case of a stand-alone software application, the copy pirate is able to sell the software after investing only of few cents for a blank CD and booklet, indistinguishable from the original, and – way below the market price. This has become a very lucrative business model which is, according to statistics, extremely wide spread in Asia, but in a strict sense is a global phenomenon.

Where the software is linked with a machine or a control system, the electronic and mechanical parts are copied one-to-one. Again, the material investment for the counterfeit copy is much smaller than the development costs for the manufacturer. Often, the machine owner who uses the pirated copy does not even notice that it is not an original machine, because next to features, the branding is also copied. The manufacturer does not find out about the pirated copy until the machine owner has a support ticket, and is not listed as a customer.

Next to pirated copies, outright plagiarized products are also a bigger-than-expected problem. In this case, the know-how is extracted from the product by the competition using reverse engineering, and then reused in their own product. This drastically reduces the R&D costs of the competition, and helps them to catch up a market lag with a giant leap.

The more that code segments are written in programming languages which generate a readable temporary code, e.g. .NET or Java; the easier it is to reverse engineer the completed application. Using state-of-the-art tools re-readable source code is generated from the application. But also applications written on C. C++ or Delphi are not immune to reverse engineering.

Logically, it is essential that product-integrated R&D efforts get protection against counterfeiting and plagiarism. This is critical for a company that wants to keep its market edge, and realize its sales potential.

2 Software Protection at a Glance

With the *CodeMeter*® / *CodeMeterAct* software protection and license management system, you can safely protect your software against both illegal copying and reverse engineering from your competitors.

WIBU-SYSTEMS has developed *CodeMeter* / *CodeMeterAct* as a standard system combining hardware-based protection (dongle - *CodeMeter*) and software-based protection (activation - *CodeMeterAct*). This means that you can ship your software as a single application, and decide later whether or not to use the dongle, use activation, or use both on a customer by customer basis.

If implementing *CodeMeter* / *CodeMeterAct*, into your software, and integrating them into your Back-Office workflow is a major concern. Then you will be happy to know about our comprehensive tools, easy-to-use APIs, and that we provide very personal support.

Figure 1: Software Protection with *CodeMeter* / *CodeMeterAct* at a glance

Security

With state-of-the-art encryption algorithms (AES, ECC and RSA) coupled with alternating keys, *CodeMeter* / *CodeMeterAct* will provide maximum security, field-tested in several hacker's contests. (See http://www.hackers-contest.com).

Licensing Models

Moreover, the *CodeMeter* / *CodeMeterAct* System is sophisticated enough to cover even the most outlandish licensing model: from single-user to network overflow licenses.

License Management

To facilitate the management of generated and shipped licenses, The *CodeMeter* / *CodeMeterAct* System provides a database-supported solution which easily integrates into existing systems, as well as comprehensive APIs to meet individual requirements.

License Activation

Four ways exist to securely activate licenses: program a dongle locally, program a dongle remotely, automatic or manual activation over the Internet, or activation by telephone.

3 CodeMeter / CodeMeterAct

3.1 A License Server for all cases

The basic component of the *CodeMeter* / *CodeMeterAct* System is the *CodeMeter License Server* (CodeMeter.exe) which runs on each computer where protected software is to be used. This component is available for different platforms. The *CodeMeter License Server* provides the interface between your software, and the license in a dongle (*CmStick*) or in an activation file.

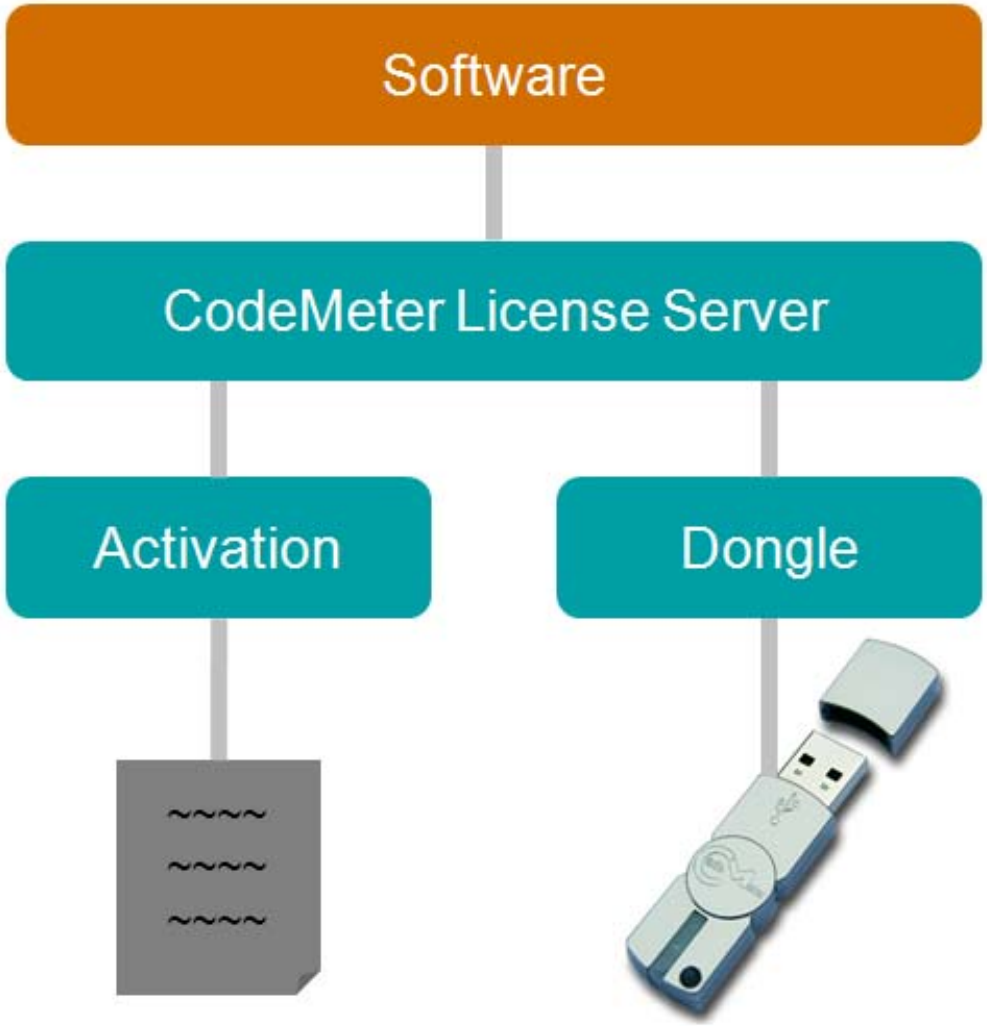


Figure 2: Standard License Server for *CodeMeter* / *CodeMeterAct*

Selectively, you can use the *CmStick*, activation, or both. However, in this context, the *CmStick* offers you superior security because all encryption and decryption is safely handled inside the *CmStick*. In addition, *CodeMeter* gives you the option to totally delete a license on the *CmStick*, or to lock the *CmStick*, when attempted fraud is detected.

3.2 The *CodeMeter* Concept

Many different software developers can store license entries in a *CmStick*, or in an activation file. Up to 4,000 license entries can be stored inside the *CmStick* and the number of licenses in the activation file is limited only by end-user disk space. These license entries consist of a FIRM CODE (FC), a PRODUCT CODE (PC), and separate feature-set parameters. The FIRM CODE is assigned by WIBU-SYSTEMS and the PRODUCT CODE is freely defined by the software developer.

Each unique FIRM CODE is assigned *once* by WIBU-SYSTEMS, and is bound to a master dongle, the so-called FIRM SECURITY BOX (FSB).

In this way, WIBU-SYSTEMS ensures that only you, as the owner of the FIRM SECURITY BOX, are able to program *CmSticks* or generate an activation files using your unique codes. Programming or activation is protected by cryptography. The encryption keys needed are safely stored in your unique FSB.

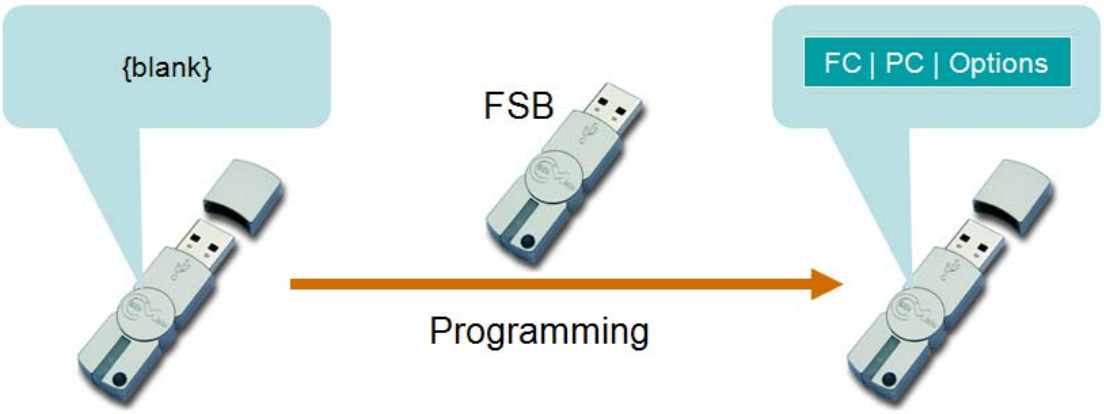


Figure 3: Programming a *CmStick*

For a better understanding of the *CodeMeter* architecture, think of each *CmStick* as a very tall file cabinet. Each software developer has access to one drawer in the cabinet into which he can add as many folders as he likes. Each license entry can be thought of as a folder in the developer's drawer. Each folder contains the (same) FIRM CODE, a unique product code and option codes. In our *CodeMeter* documentation, the file drawer is called a "FIRM ITEM" (in the sense that Firm = Company). Each "folder" or license entry is called a PRODUCT ITEM.

A *CmStick* is able to simultaneously hold license entries from many different software or content developers. Thus, several development companies can share one *CmStick*, saving hardware and shipping expenses. At the same time, the end-user benefits by having only one dongle to store all licenses. This gives the end-user the important benefit of license portability and also provides a way to securely backup and restore his important licenses.

Needless to say, in the case of *CodeMeterAct*, several license files from different developers may be installed simultaneously.

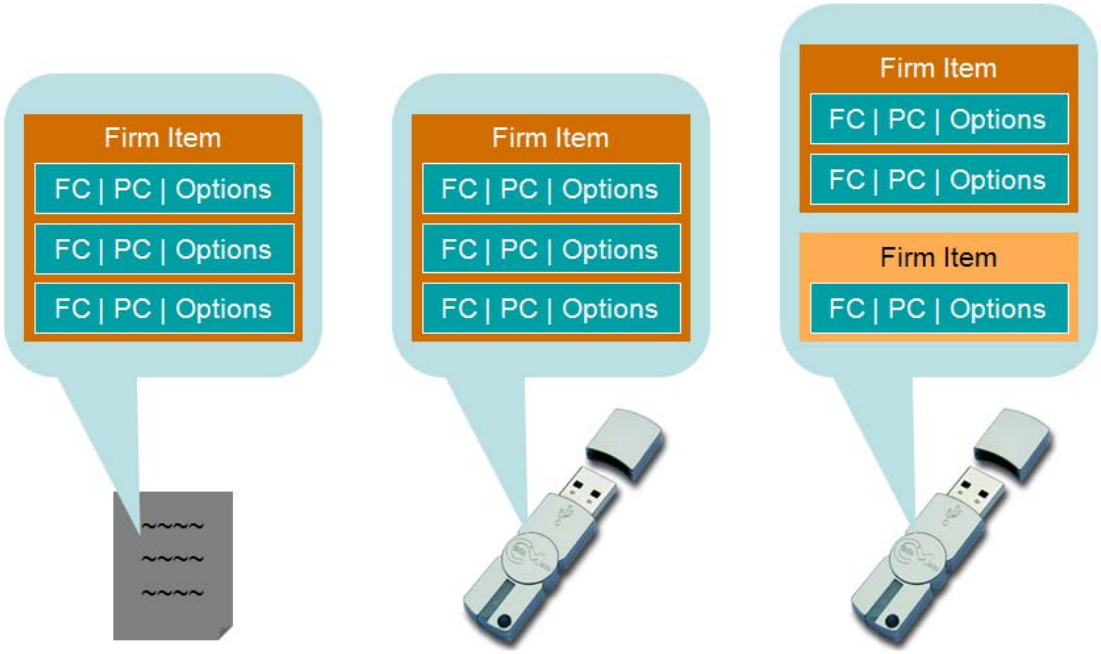


Figure 4: Activation file with one FIRM ITEM, *CmStick* with one FIRM ITEM, *CmStick* with two FIRM ITEMS of different developers

Since each license entry can hold several combinable options (PRODUCT ITEM OPTIONS) a developer is able to define individual license models for each customer. *This is an important feature that can save the developer lots of time and money.* Why? Because the developer no longer needs to spend time altering installations on a customer by customer basis. Instead, all customers receive the same software and the license options are defined in the *CmStick*, or in the activation file.

In order to modify these feature-set options, your FIRM SECURITY BOX is required. In this way, we ensure that your customer is not able change the license you sold. Only the options 'Text' and 'User Data' can be modified without a FIRM SECURITY BOX. Table 1 shows the properties of the single options:

Product Item Option	Description	Read Access	Write Access
UNIT COUNTER	Counter, use for pay-per-use, pay-per-click, pay-per-print or pay-per-start versions	yes	reduce, yes / increase with FSB
LICENSE QUANTITY	Number of simultaneously available licenses, use for network floating licenses	yes	with FSB
ACTIVATION TIME	Activation date, use for temporary limited versions	yes	with FSB
EXPIRATION TIME	Expiration date, use for temporary limited versions	yes	with FSB
USAGE PERIOD	Usage duration, use for temporary limited licenses for which the starting date has yet to be defined (demos, backups)	yes	initially at first start
FEATURE MAP	32 bit map, use for activation of features or for version management	yes	with FSB
CUSTOMER OWNED LICENSE INFORMATION	256 byte text field, use for customer-specific data (e.g. name of user)	yes	with FSB
TEXT	256 byte text field, use for display in WebAdmin	yes	yes
USER DATA	256 byte data, use for saving configuration data	yes	yes
PROTECTED DATA	128 x 256 byte data, use for saving additional information in binary format	yes	with FSB
HIDDEN DATA	128 x 256 byte data, use as key source	with password	with FSB
SECRET DATA	128 x 256 byte data, use as key source	no	with FSB

Table 1: PRODUCT ITEM OPTIONS Overview

CodeMeter WebAdmin allows the user to view the license entries in a *CmStick*. *CodeMeter WebAdmin* is integrated into the *CodeMeter License Server*.

Figure 5: *CodeMeter WebAdmin* example – two different licenses

The programming of the *CmStick* or the information in the license file define the license models. Thus, license models are separate from the protected software, and can subsequently be modified.

3.3 Operating Systems Supported by *CodeMeter* / *CodeMeterAct*

CodeMeter / *CodeMeterAct* are available for many operating systems:

Operating System	*CodeMeter*	*CodeMeterAct*
Windows 2000	yes	yes
Windows XP	yes	yes
Windows Vista	yes	yes
Windows 2000 Server	yes	yes
Windows 2003 Server	yes	yes
Windows 2008 Server	yes	yes
MacOS X	yes	yes
Linux	yes[1]	yes[1]
Sun Solaris 9	yes	-
Sun Solaris 10	yes	-
Windows XP Embedded	yes	-
Windows CE 5.0	yes	-
Windows CE 6.0	yes	-

Table 2: Operating Systems Supported

[1] See release list for detailed information on Linux distributions supported.

3.4 *CodeMeter* Hardware Models

The *CodeMeter* hardware comes in various forms for different interfaces:

CmStick Standard version in an attractive plastic shell

CmStick/M Standard USB version with additional flash memory allowing applications to start directly from the *CmStick*

CmStick/ME Metal USB edition, in a polished chrome metal shell, optional with flash memory

CmCard PCMCIA card edition, with flash memory

CmCard/E ExpressCard edition, with flash memory

CmCard/SD SD Card, planned for the future

CmASIC Use our ASIC on your own board to protect embedded applications

Figure 6: Selection of *CodeMeter* Hardware Models

3.5 *CodeMeterAct* Binding Schemes

Licenses in a *CodeMeterAct* license file are bound to a PC. Specific properties of the PC or the installed OS are used to build up information used as a "Host-ID". This Host-ID becomes part of the activation code. The activated license can only be used on the PC if the Host-ID has not changed significantly since its activation date.

CodeMeterAct's Host-ID scheme makes use of both Hardware Binding Schemes (real hardware properties) and Configuration Binding Schemes (configurable system properties).

3.5.1 Hardware Binding Schemes

Four basic hardware properties are available; which can be combined, without restriction, to create the Hardware Binding Scheme:

Hardware Property	Description
Network-Adapter (N)	Information on the network card (MAC Address), virtual adapters are not considered
Disk (D)	Information found on the hard disk (real serial number)
CPU (C)	Information found on the main processor (processor type)
Bios (B)	Information found in the PC BIOS

Table 3: Basic Hardware Properties

By altering the combination of properties and by changing how many properties need to remain unchanged, the developer can control how restrictive to make this binding scheme. More restrictive schemes increase security, but also increase the number of support calls from customers who might have simply changed a hard drive (for example). The less restrictive, the fewer support calls, but security becomes weaker. For example, a scheme combining all four properties, where only one element is allowed to change, is possible. As long as three elements remain unchanged, an activated license remains valid.

The following table shows some recommended schemes:

Scheme	Description	Use
N:1	Binding to a network adapter.	Big enterprises, SMEs (Small and Medium-sized Enterprises)
DCBN:4	Binding to network adapter, CPU, hard disk and BIOS. Zero tolerance towards hardware changes.	Small Office / Home Office, Games
DCBN:3	Binding to a network adapter, CPU, hard disk and BIOS. Three properties have to remain unchanged, one is allowed to change.	Big enterprises, SMEs, Small Office / Home Office
DCN:3	Binding to BIOS, CPU and hard disk. Zero tolerance towards hardware changes.	Big enterprises, SMEs, Small Office / Home Office
DCB:2	Binding to BIOS, CPU and hard disk. Two properties have to remain unchanged, one is allowed to change.	Big enterprises, SMEs, Small Office / Home Office
D:1	Binding to hard drive	Small Office / Home Office

Table 4: *CodeMeterAct* Hardware Binding-Schemes

3.5.2 Configuration Binding Schemes

Compared to hardware properties, binding to configurable properties is considerably weaker. In addition, the properties cannot be combined:

Scheme	Description	Use
IP-Address (IP)	The IP-Address of the PC is part of the Host-ID calculation. The IP-Address is used to which the CodeMeter License Server is linked.	Big enterprises
Machine-SID (MID)	The Machine-SID and the Domain-SID in the Windows network are part of the Host-ID calculation.	Big enterprises
None (Non)	The license file is not bound to a PC, and able to be activated with a fixed, pre-calculated code on any number of PC. For license-related reasons, this mode works with time-restricted licenses only. This scheme is suitable for demo versions without online activation (specification of product serial number and activation code on the package only). The activation code is the same for all installs. After the license has expired, it cannot be used again on the same PC.	Small Office / Home Office (Demos)
Serial (Ser)	The product serial number is part of the Host-ID calculation, i.e. the Host-ID is different for each separate user. However, the Host-ID does not change with the change of computer hardware. This scheme is designed as a simple license protection. Additionally, the developer is able to write the name of the user into the license file; read it; display it onscreen; or print it. That increases the perceived individuality of the license, and raises the mental barrier to pass on the license illegally. In this scheme, activation codes can be pre-calculated, i.e. the user gets a product serial number and an activation code, enters both, and the software runs without online activation.	Big enterprises

Table 5: *CodeMeterAct* Configuration Binding Schemes

The binding scheme is defined independently from the software, and is subsequently editable for software already encrypted with *CodeMeterAct*.

The end-user receives a license information file holding information on the scheme used. This license information file may be shipped together with the software, e.g. as part of the installation, or sent separately to the end-user. This way, the developer is able to individually define the scheme for each installation.

For example, the developer can produce a CD which is the same for all customers but contains a license information file with the scheme `DCBN:3`. This file is sent to all medium and large customers. In the event where a large wants customer wants to renegotiate his license terms. For example, covering the company-wide use of 100 licenses excluding online activation, then he simply receives a new license information file with the

license bound to the IP-Address or Product Serial Number, and a list with 100 product serial numbers and activation codes.

4 License Models

Each license entry can hold many combinations of options; providing the software developer with the ability to map numerous license models with *CodeMeter* / *CodeMeterAct*. The following table shows some of the possible license models.

License Model	Description
Standard license	Created from an entry consisting of both the Firm Code and Product Code. Depending on the implementation in the software, the license can be on the local PC, or can be on a network license server. In addition, the developer can define (in the software) if the licenses should run concurrently.
Local license	Created from an entry consisting of the Firm Code, Product Code and with License Quantity = 0. The *CmStick* or the activation file should reside on the same PC where the software is installed. When operating under VMware, the license has to be directly available within the session. Sharing between different sessions is not possible.
Floating license	Created from an entry consisting of the Firm Code, Product Code and with License Quantity = x. With x you set the number of concurrent licenses. You are also able to select if the count should decrement per PC or per instance.
Time-limited license (fixed date) Demo version leasing license	Created from an entry consisting of the Firm Code, Product Code and with an Expiration Date and/or Usage Period set. The time period can be absolute or relative. In CodeMeter the Expiration Date or Usage Period is compared with the internal clock in the *CmStick*, and thus is protected against manipulation. In *CodeMeterAct* the Expiration Date is compared to the PC clock, and has sophisticated algorithms that can detect if the system clock has been manipulated.
Pay-per-use license Pay-per-click license Demo version (x-times starting)	Created from an entry consisting of the Firm Code, Product Code and with a Unit Counter. You define when the Unit Counter counts down. In an "x-times starting" version this takes place at application start. In the case of a pay-per-use or pay-per-click license you set the Unit Counter forward, or down after the related action in your application. If you want, different actions, can count down the same counter or a different counter.
Runtime-limited license	Created from an entry consisting of the Firm Code, Product Code and with a Unit Counter (value = runtime / time unit). In the software you count down the Unit Counter by 1 per time-unit.

Functional-limited demo version	Created as several entries from the FIRM CODE but with different PRODUCT CODES.
Modular license	Each PRODUCT CODE stands for a module or functionality. By programming the related PRODUCT CODES you are able to generate an individual license for the user. In this case, you are able to separately add further options to each module, e.g. EXPIRATION DATE. OR Created from an entry consisting of the FIRM CODE, PRODUCT CODE and with a FEATURE MAP. Each bit in a feature map exactly represents a module or functionality By programming the related FEATURE MAP you activate the single module or functionality. The licenses of the single modules may be spread over different *CmSticks* or license files. For example, the basic version runs with PC binding, while the support technician gets extended access to other functions with the CmStick.
Downgrade license	Created from an entry consisting of the FIRM CODE, PRODUCT CODE and with a FEATURE MAP. Each bit in the FEATURE MAP stands for a different version. For example, you are able to offer simultaneously a floating license on 3 PCs including a downgrade privilege, i.e. the user is able to run either the old or the new version on 3 PCs, but both versions simultaneously on a maximum of 3 PCs.
Cold-standby license	Created from an entry consisting of the FIRM CODE, PRODUCT CODE and with a USAGE PERIOD. The backup dongle is set with a time period you define - beginning with first use, e.g. 7 days. Within this time, the user should have enough time to replace the original license (dongle) and reset the standby dongle for future use.
Hot-standby license	Using the server search list you define the sequence of the licenses to be applied. In the event the first server fails, automatically the second server with the backup licenses is used. You monitor the counter reading on a regular basis to avoid misuse.
Overflow license	Created as two entries from the FIRM CODE and two different PRODUCT CODES. The main entry does not hold a UNIT COUNTER but has a LICENSE QUANTITY according to licenses acquired. The overflow entry holds a very high UNIT COUNTER and includes a LICENSE QUANTITY entry with the number of desired overflow licenses. In the event that all main licenses are used, you use the overflow entries in the software. Then you are able to decide on your own whether to display this in the software, and in this case artificially slow down software performance. Additionally, you are able to monitor the counter reading on a regular basis, and interpret how often (and how long) the overflow licenses have been used.
PC bound license	Created with *CodeMeterAct* and the related binding scheme (hardware or configured machine) or alternatively, as an entry from FIRM CODE, PRODUCT CODE and includes saving a separate Host-ID as PROTECTED DATA. In the software you compare if the saved Host-ID matches the actual calculated Host-ID for the PC.
Named user license	Created with *CodeMeterAct* and the related binding scheme (user) or alternatively, as an entry from FIRM CODE, PRODUCT CODE and with saving a separate User-ID as PROTECTED DATA. In the software you compare if the saved User-ID matches the actual calculated User-ID for this user.

Table 6: License Models

All license models are freely combinable. Multiple entries – with identical FIRM CODE and PRODUCT CODE – are totaled at a license server. For example, you are able to add further floating licenses to an existing floating license for just a certain period of time.

Figure 7: Eight Floating Licenses for „Product One", spread over two license items

5 License Activation

5.1 Programming a *CmStick*

For programming a *CmStick* you require a FSB which holds your FIRM CODE (see 3.2 The *CodeMeter* Concept).

You program a *CmStick* in the following ways:

- **directly** (the *CmStick* is programmed by the developer)
- **by data transfer** (the *CmStick* is already with the user and is remotely re-programmed)

In the case of direct programming, the FSB is connected to the local PC or to a network license server. Then access control on a network server assigns the local PC privileges to access the FSB.

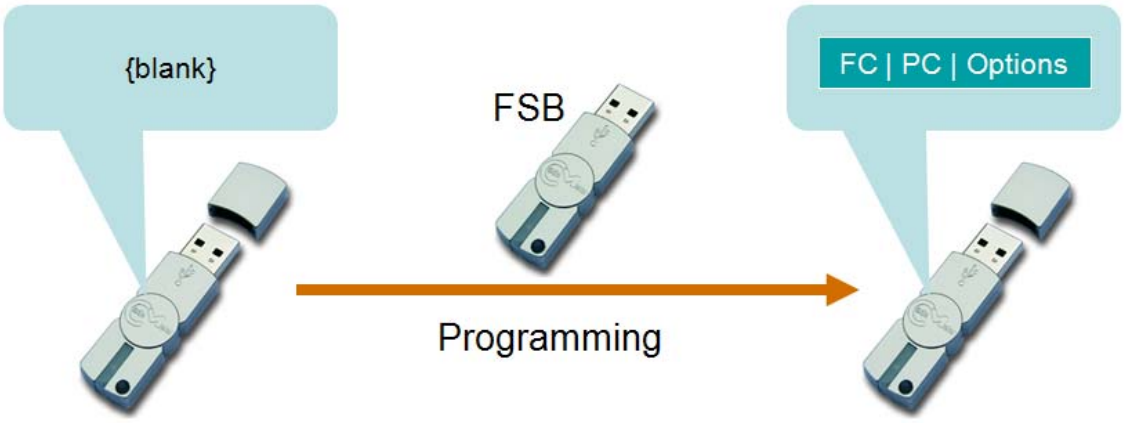


Figure 8: Programming a locally connected *CmStick*

As Figure 8 shows with the FIRM CODE in the FSB you are allowed to program a license entry in an empty *CmStick*. Of course the *CmStick* to be programmed may already hold other license entries. In this case, it does not matter whether these are the developer's own license entries (with the same FIRM CODE) or license entries from other developers.

In the case of programming by file transfer, a context file is created. Again, it does not matter whether the *CmStick* is empty, or has existing license entries. The context file contains the serial number of the *CmStick*, and the content of the developer's unique FIRM ITEM and represents the license container summing up all of the developer's unique license entries.

When the developer creates an update file using the FSB; this update file can be used only once on the designated *CmStick* by the user. After successfully restoring the update file, a counter in the FIRM ITEM is incremented. By incrementing the counter, the update file becomes invalid for a further restoration. This is particularly relevant in case where the update file holds programming commands which create a license item, which increment a UNIT COUNTER by a number of units, or forward an expiration date by a number of days.

Locally connected *CmSticks* can be reprogrammed either manually with the manual exchange of update files or via TCP/IP with an automatically created update file.

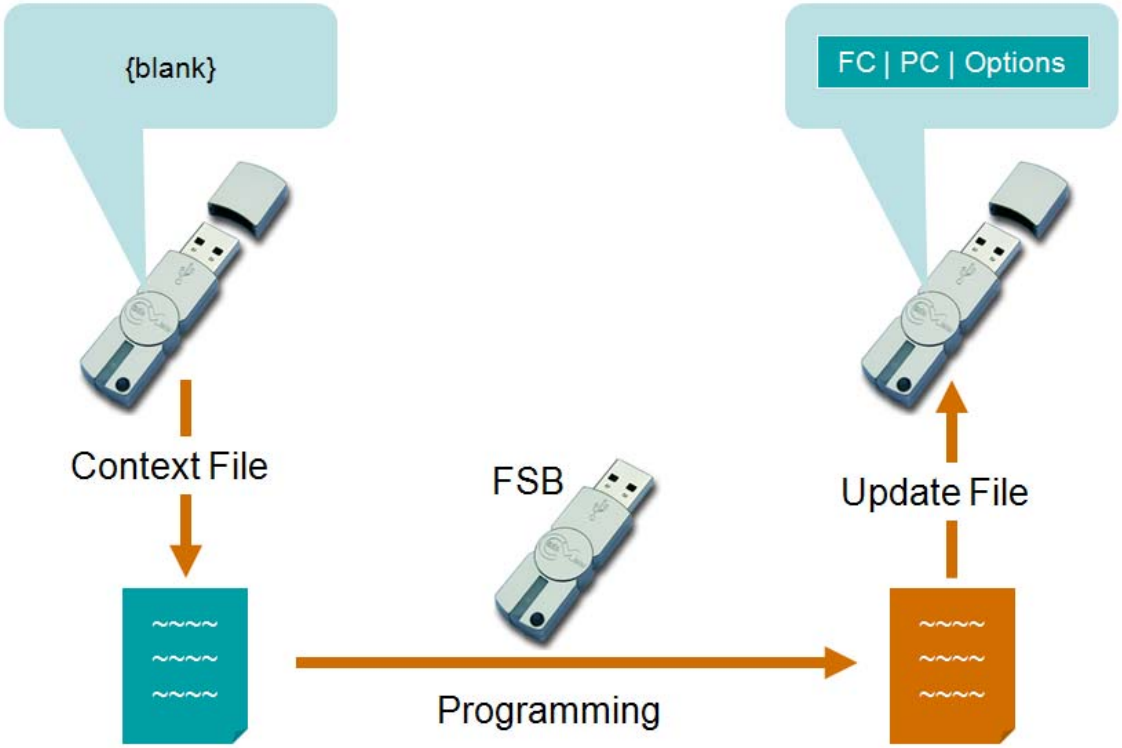


Figure 9: Programming by Update File

In order to ease the reprogramming process, the developer has the ability to generate the next context file, and store it as a modified context file on an Internet server. The generation of and sending of the next context file is triggered by the end-user. The developer's server then sends the update files only. This is feasible, because only the status of their own FIRM ITEM is relevant for programming the developer's own license entries, i.e. the counter in the FIRM ITEM. In the case where the *CmStick* is, in the meantime, re-programmed by another developer, the update files, context files and modified context files stay valid.

In the offline case, the context file is automatically generated and uploaded by the user, the update file is downloaded and stored, and a new context file is generated as and uploaded. Thus, the developer has an actual overview of the licenses already stored with each individual customer.

You are able to integrate this functionality into your software, or you can use WIBU-SYSTEMS' standard software installed with the *CodeMeter* runtime. See chapter 8 License Management - Backoffice Integration for a detailed description of the online module designed for developer use.

5.2 Activation of a *CodeMeterAct* License

5.2.1 Automatic Activation

In the case of *CodeMeterAct*, instead of a *CmStick*, a license information file is delivered. This file is comparable to an empty *CmStick*, and holds the information of the *CodeMeterAct* Binding Scheme.

When automatically activated, a *CodeMeterAct* license (a context file) is generated from the license information file, the product serial number, and the Host-ID.

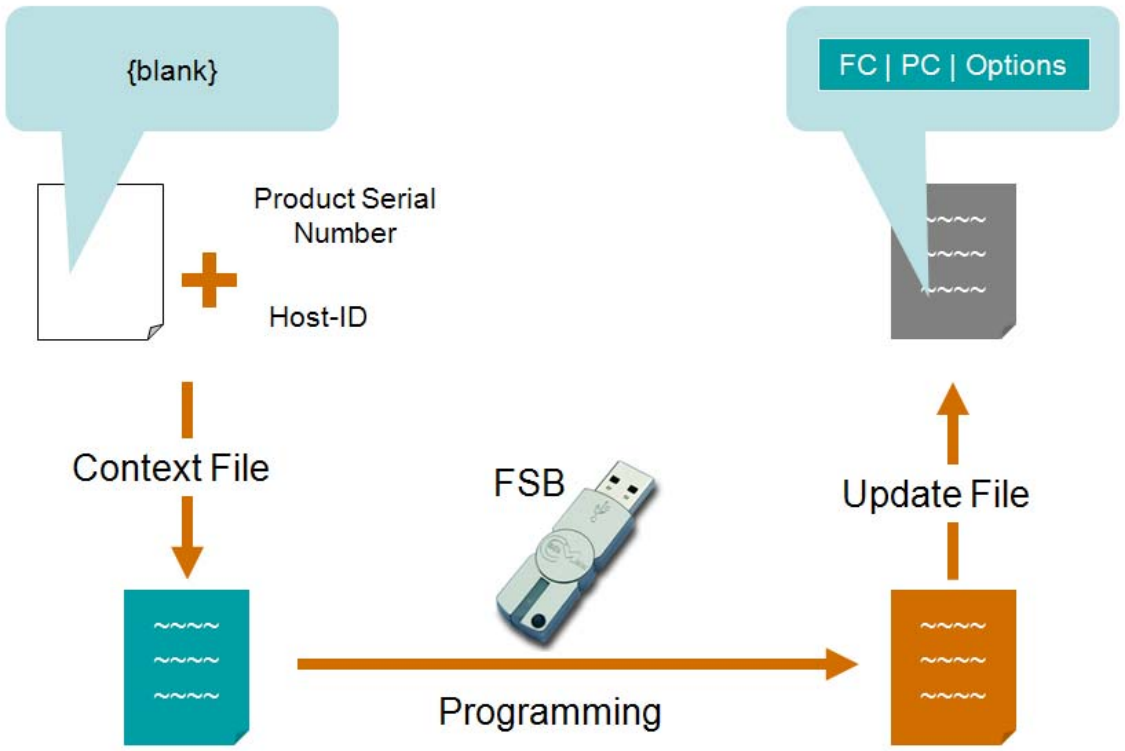


Figure 10: Automatic Programming of a *CodeMeterAct* License

The context file generated then is sent from the end-user to the developer. For this the following options exist:

- **Directly from within your software by SOAP:** You automatically generate the context file and install the update file. You are also able to easily integrate the activation using an application automatically protected by *AxProtector* using an error message interface (dll).
- **Directly over a website:** An installed *CodeMeter* runtime is required, the user accesses the website, and the generation of the context file and the installation of the update file are done automatically.
- **Indirectly over a website:** The context file is generated by the user and is uploaded; afterwards the new update file is downloaded and manually installed by the user. This allows activation of a license on a PC which does not have internet access.

5.2.2 Activation by Telephone

The automated activation requires internet access, however, not necessarily on the PC where the license has to be activated. But there must be a way to transfer data by files between this PC and another PC with internet access. In case this requirement is not met, WIBU-SYSTEMS offers activation by telephone.

In contrast to the automatic activation, the license information file is not delivered empty, but holds a large part of the necessary information. Only a secret missing part, of the cryptographic keys, is transferred on activation.

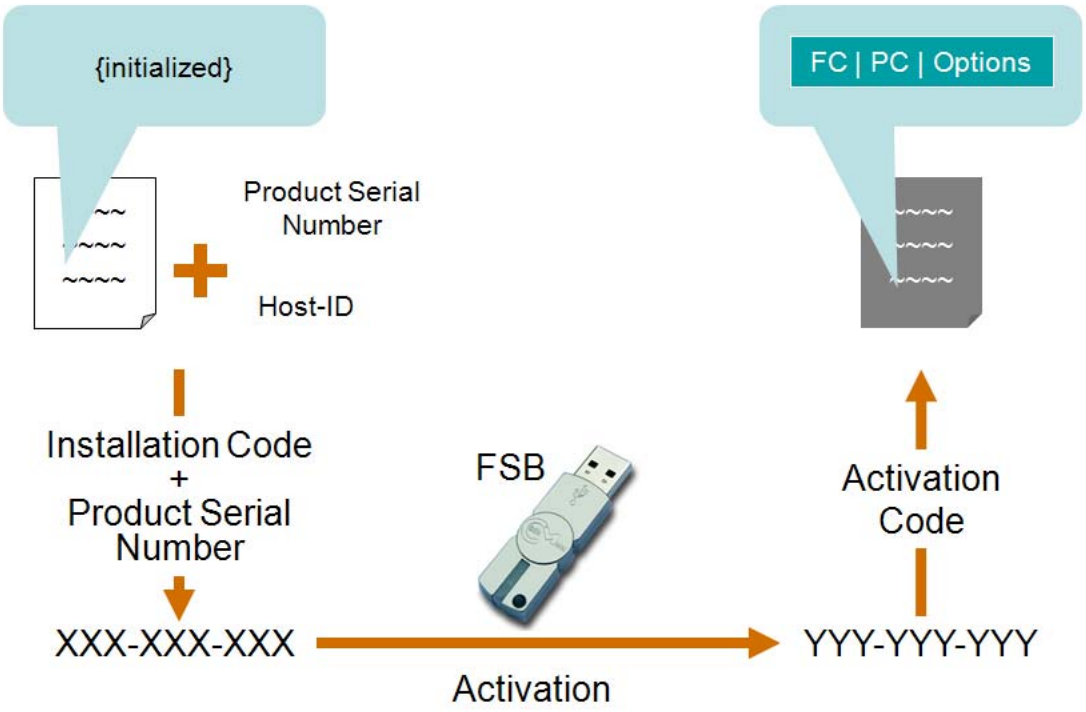


Figure 11: Activation of a *CodeMeterAct* license by telephone

With "activation by telephone", an installation identification number (hash) is created for the existing license information file on the user's PC and consists of the Host-ID and the product serial number. The end-user informs the developer's help line of both this installation identification number and the product serial number over the telephone.

When using the *CodeMeter License Central* (see chapter 8.2 *CodeMeter License Central* – The Core) the developer calculates the activation code, recites it to the user, who then types it into the application activation window. The license is then activated.

If a developer wants to implement the automatic activation or activation by telephone he is provided with API functions to integrate activation into his software and into his design. For the automatically protected application (see 7.1 *AxProtector*) you will find an implementation example in the error message dll.

Developers are also able to automate the "activation by telephone". Here, a voice-controlled computer or a web page is used instead of a person. In this case, the end-user types in the installation identification, and receives the result by telephone or display.

When using activation by telephone you have two basic options for coding the installation identification and the activation code:

- Alphanumeric mode: use of 40 alpha-numeric characters (0 .. 9, A..Z), length: 40 characters
- Numeric mode: use of 10 numbers (0..9), length: 60 characters

Using an API function you are able to switch between the two options. The numeric code option requires a greater length but has the advantage that the installation code is easy to enter with telephone keys when a voice-controlled computer is linked. The numeric telephone transfer of the activation code is also easy because internationally 10 numbers are easier to spell than alphanumeric characters.

5.3 Multiple Activations of the Same License

It is very rare that a *CmStick* fails. But, in order to avoid any customer inconvenience at a mission critical site, a backup *CmStick* holding all licenses with the defined usage periods is pre-accorded to the customer. By contrast, it is almost a foregone conclusion that a PC on which a license is activated will get updated or replaced quite frequently. Here the recovery of the initial state requires the activation of a license for an additional Host-ID. Based on the developer's marketing experience, one activation per year in case of non-tolerant schemes may be required irrespective of business or consumer products.

However, when configuring the license (see chapter 8 License Management - Backoffice Integration) the developer has the option of specifying how often the related license is permitted to be activated. When the number of permitted activations is reached, the developer's support staff is able to individually increment this number for each acquired license. Subsequently, there is no limit to reactivation.

Reactivating for the same product serial number and the same Host-ID is not viewed as multiple activations, and so, is also feasible when the number of permitted activation has already been reached. In the case where this product serial number is already activated on several Host-IDs, it is only valid for the last known Host-ID matching this product serial number.

In order to limit the abuse of multiple activations, the developer has the option to time-limit licenses in different configurations:

- **Expiration date**: An expiration date is set in the license. The license then automatically expires on this date, and has to be reactivated. The developer is able to integrate a license renewal into the software as an automatic update. In this case, the user sees the expiration date in the *CodeMeter WebAdmin*. This procedure could be used to facilitate the leasing of licenses where the user should know the leasing period.
- **CheckPoint**: In the license a timed CheckPoint in an array is defined. Either as plain text ("Next update on 08-31-2008") or encrypted and hidden. The software automatically scans for the CheckPoint. When the Checkpoint is reached, the software tries to reactivate the license, the old CheckPoint is deleted, and the new CheckPoint is set. The error handling ("License could not be reactivated") completely resides in the software.

In both cases, the license is automatically reactivated based on the software. However, in the case where the Host-ID does not match the current Host-ID for this product serial number and the number of permitted activations is reached, a reactivation of the license is not permitted. When an error occurs, the support staff is able to increase the number of activations, or individually change the binding scheme for this particular customer.

6 Security

6.1 The Use of Encryption

The security of *CodeMeter* / *CodeMeterAct* is based on encryption. The software to be protected or modules or data in the software are encrypted by the developer before shipping. The key for decryption is part of the license the developer generates for the end-user. On the user's side, parts of the software are decrypted only when needed. After use, these parts can be re-encrypted.

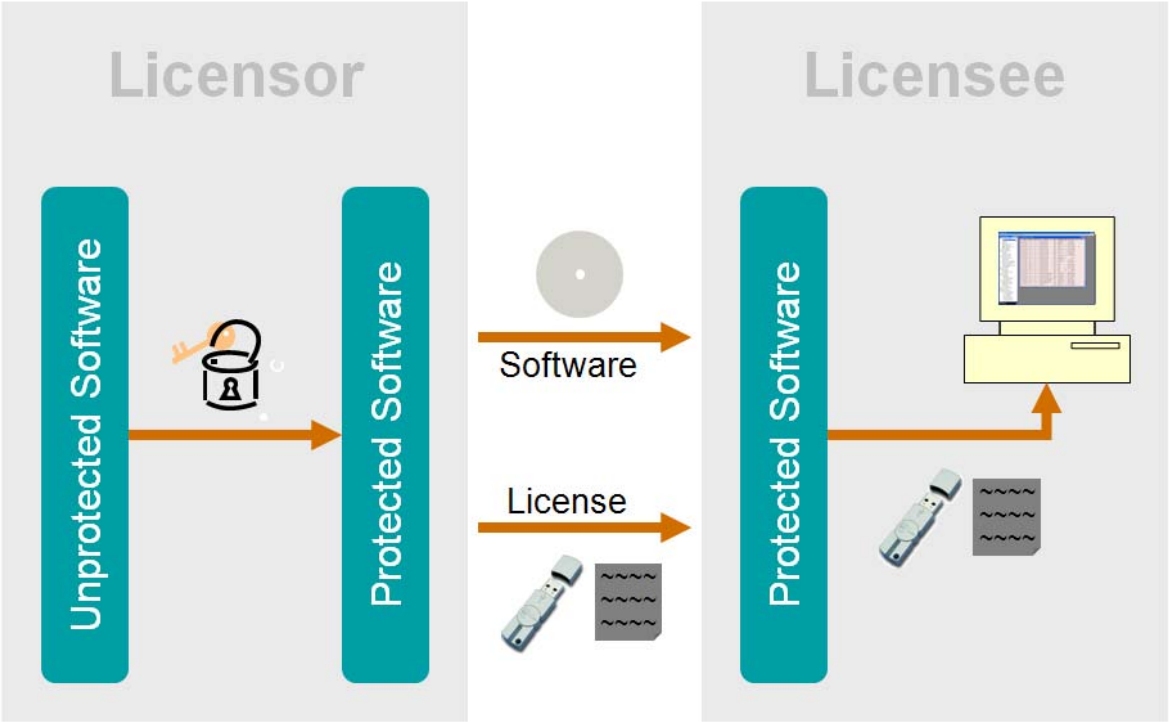


Figure 12: Security by Encryption

CodeMeter and *CodeMeterAct* are based on the same principles and map the same license models. In *CodeMeterAct* (which completely runs in PC memory) different secret keys are used than those used by *CodeMeter*, i.e. analyzing *CodeMeterAct* does not help a hacker draw any conclusions for the *CodeMeter* keys safely stored in the smartcard chip.

6.2 Hardware Advantages

CodeMeter proved its field-tested security in several hackers' contests. A music software manufacturer who decided to use *CodeMeter* said, "6 months of security life was our objective, and now, after more than 3 years there are no cracks to our software. This exceeded our expectations by far".

The strengths of *CodeMeter* are clearly apparent. The following table shows the security-relevant aspects where the dongle solution is far superior to the activation solution.

CodeMeter Advantages	Description
Firmware runs protected in the hardware	The firmware, i.e. key storing and calculation, and the related encryption and decryption are safely protected and run in the *CmStick's* smartcard chip. The hacker cannot analyze the chip because it represents a black box.
Hardware is able to be locked	In case you detect an attack within your software (this is done automatically by our tools), you are able to send a lock command to the *CmStick* directly from within your software. This command locks all your licenses, i.e. those in your FIRM ITEM. You are able to reactivate these licenses by remote programming. However, until reactivation the *CmStick* behaves as if those licenses (and the keys involved) were not present. The hacker does not have a second try.
Counters cannot be set back by a backup	Counters are safely stored in the *CmStick's* smartcard-Chip. The "counter values" cannot be manipulated from the outside and cannot be reset by installing a backup.
Deleted licenses cannot be set back by a backup	Licenses which have been deleted in a *CmStick* no longer exist. By transfer of a receipt, the developer is sure that the license does not exist in the current *CmStick*, and also is irretrievable.
Expiration Time and Usage Period are checked against the internal clock	All times and dates used, such as, EXPIRATION TIME and USAGE PERIOD are checked against the clock running internally in the smartcard chip. The recorded times cannot be manipulated; the internal clock cannot be set back. Consequently, an expired license is irretrievable. For futher security, the developer can update the internal clock via a *CodeMeter Certified Time Server*.
License Portability	The user wants the convenience of using software legally purchased on different computers (home, office, etc). The developer wants to make sure that his programs are not used illegally on multiple computers. With *CodeMeter*, both the user and developer are winners; since the license is contained on the *CmStick*, the user can move it by simply relocating the *CmStick*. And the developer knows that while his program may be installed on more than one system, it can only be used on one of them at a time.
Security against license loss by viruses and other malware	Programming (create, edit, delete) of a license in a *CmStick* is secured by cryptography. Only you with your FSB are able to delete entries. No virus is able to destroy the user's licenses.

Table 7: Hardware Advantages

6.3 One License Entry – Many Keys

The software is encrypted at runtime on the user's PC. At runtime the communication between the software and the license is encrypted (in the case of *CodeMeter* even as far as into the *CmStick*). A common practice among hackers is to use a "record / playback" tool at the interface in order to discover the encryption key. This is not possible with the *CodeMeter* System because WIBU-SYSTEMS uses the concept of alternating keys.

Each license entry provides a set of 4 billion different keys to be used for the protection of the software. These keys are generated in the *CmStick* / *CodeMeterAct* module by a derivation. This means that, *CodeMeter* and *CodeMeterAct* use different secret keys.

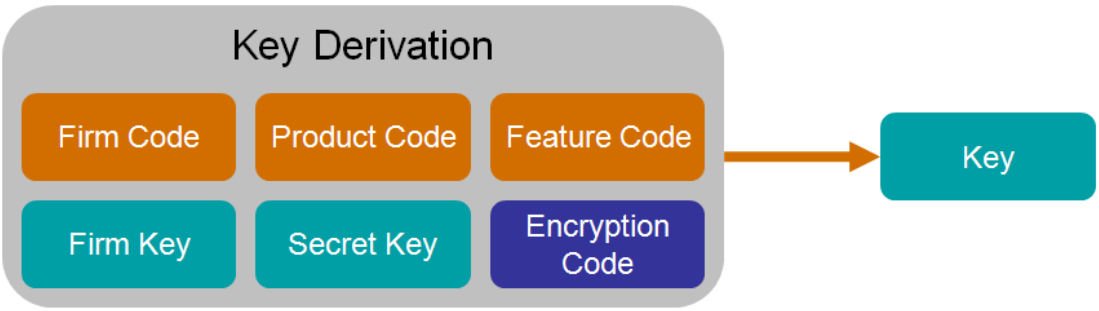


Figure 13: Key Derivation

The FIRM CODE, PRODUCT CODE and FEATURE CODE present the visible elements. You program these into the *CmStick* or the license file. The user is able to see these elements in *CodeMeter WebAdmin*. These elements define the license entry.

The FIRM KEY contains the secret element on the developer's side. Normally, the FIRM KEY is defined by WIBU-SYSTEMS. However, in *CodeMeter* the developer is able to freely define the FIRM KEY and write it to the FSB. On programming a *CmStick* or license file the FIRM KEY is transferred.

The SECRET KEY is an additional secret element of *CodeMeter* / *CodeMeterAct* which is created and used differently for both copy protection systems.

SECRET KEY, FIRM KEY, FIRM CODE, PRODUCT CODE and FEATURE CODE are fixed for a license entry. In contrast, the encryption code is modifiable at runtime. By changing the encryption code you are able to work with alternating keys.

The developer's task is to integrate the concept of alternating keys using appropriate methods. For example, by encrypting the same data with different keys before shipping, and decryption at runtime is contingent upon correct key selection.

In the case of an automated integration with *AxProtector* and the integration with *Wibu Universal Protection Interface* (see 7. Software Integration), the developer does not need to be concerned with this task. This is automatically done by the WIBU-SYSTEMS tools. And with tool updates new methods are automatically integrated on re-encrypting your software. All this is accomplished without changing the source code or recompiling your software.

6.4 Using *CodeMeter* and *CodeMeterAct* simultaneously

CodeMeter and *CodeMeterAct* are based on different keys. Thus, data encrypted for *CodeMeter* cannot be decrypted with *CodeMeterAct*. However, in order to allow a secure and flexible license management, WIBU-SYSTEMS tools work with a two-tiered encryption.

Encryption:

- A contingent key is generated.
- Data is encrypted with this contingent key.
- The key is encrypted with *CodeMeter* and/or *CodeMeterAct* (eventually with different encryption codes). These keys are encrypted an as a key pool attached to the data.

Decryption:

- The matching *CodeMeter* and/or *CodeMeterAct* license is queried.
- From the key pool a key is selected that matches this license.
- With the selected key the initial key is decrypted.

- With the initial key the data is decrypted.

This two-tiered encryption is automatically used in *AxProtector* and in the *Wibu Universal Protection Interface*. This allows a developer to generate software which optionally runs with *CodeMeter* or *CodeMeterAct*. Next to this flexibility, the two-tiered encryption also provides a considerable performance gain compared to a complete encryption by the dongle.

Of course, the developer is free to implement procedures of his own design using the API tools.

7 Software Integration

For software integration, WIBU-SYSTEMS provides three options:

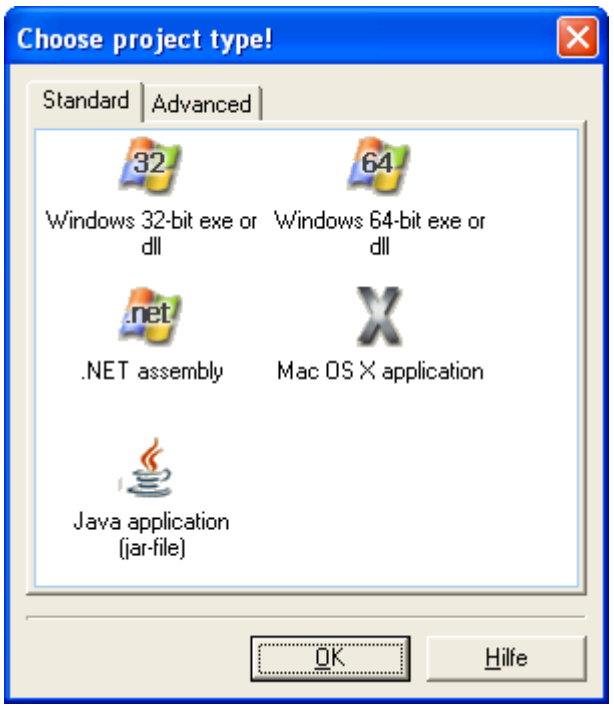
- automatically with *AxProtector*,
- individually with the *Software Protection API* [*Wibu Universal Protection Interface* (WUPI)] and the included *IxProtector* for protection at the functional level,
- with the native *Core API*.

7.1 AxProtector

AxProtector is a tool which subsequently encrypts the completed application (exe, dll, or jar).

The *AxProtector* product family comprises:

- *AxProtector* (protection for native Windows 32-bit and 64-bit applications, protection for Mac OS X applications)
- *AxProtector for .NET* (protection for .NET assemblies at the method level)
- *AxProtector for Java* (protection for Java application at the class level)



When using *AxProtector* you are supported by a graphical user-interface and command line tool for integrating your automated build processes.

In *AxProtector* you are able to select the copy protection system. You generate an application which works with *CodeMeter* only or *CodeMeterAct* only, or with both protection systems simultaneously.

AxProtector provides you a safe and very easy-to-implement protection against pirate copies and reverse engineering. You simply take the compiled application and encrypt it with *AxProtector*. *AxProtector* independently inserts code in your application which decrypts the encrypted parts at runtime. Next to decryption, this code also contains checks for application integrity, and detection mechanism for typical hacker tools.

If a hacker tool is detected, *AxProtector* immediately takes counter measures, ranging from simply shutdown of your software to permanent license locking. You are able to define for yourself for which hacker tools are detected, and how *AxProtector* is to react.

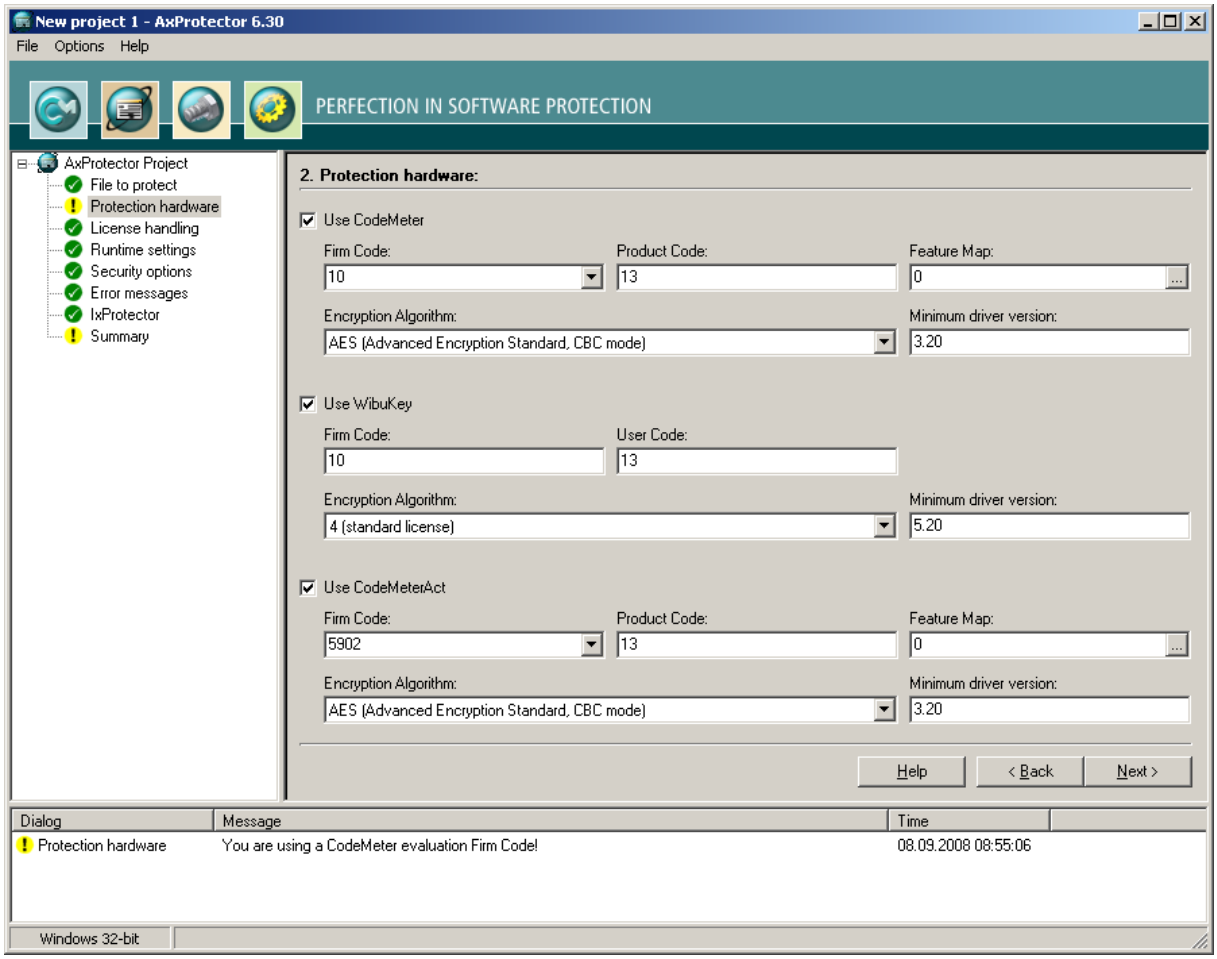


Figure 14: Selection of *CodeMeter* and *CodeMeterAct* in *AxProtector*

In addition to security, *AxProtector* also provides you many convenient functions. In a LAN environment, it can automatically reserve the related license on the license server upon the start of the application, and releases the license on application exit. With a configurable runtime check (interval) *AxProtector* ensures that your software is running only when the license is permanently located on the PC, i.e. when the dongle is not detached.

Even in error handling, you are free to decide how your application is to react. You are able to define whether error messages display and have control over the message text. Using a dll interface you are able to implement error messages yourself. This allows you, for example, to realize the activation of a license, or the programming of a *CmStick* using your own design with favored sequences in an automatically protected application.

7.2 *Software Protection API - Wibu Universal Protection Interface* (WUPI)

AxProtector provides you with a comprehensive protection for most protection scenarios. However, there will be times when AxProtector just won't suffice. In those instances, WIBU-SYSTEMS offers you an easy interface: The *Wibu Universal Protection Interface* (WUPI). The use of WUPI is recommended in the following application scenarios:

- Protection of single modules of your software (an executable file) that need to be licensed separately. Since *AxProtector* completely encrypts an executable file. You will need to use WUPI to protect additional modules which require separate licenses and the separate error handling.
- If you need to count down a UNIT COUNTER at a specific place in your software for actions (pay-per-use, pay-per-print, etc) that occur other than when the application is started. WUPI offers an easy integration tool. *AxProtector* optionally counts down a UNIT COUNTER on application start. But, if you want this to happen when an event occurs (other than application start) e.g. you want to get paid every time your customer uses your software to print a report, again WUPI offers easy functions.

- Should you want to increase the security of your application by determining for yourself which parts of your software are encrypted and decrypted and when this takes place. Then here again, you are able to bank on AxProtector security, because WUPI and *AxProtector* can be used simultaneously.

WUPI is available for many programming languages and platforms. WUPI contains easy function calls encapsulated in a dll. While implementing WUPI you work with IDs for single modules, i.e. you do not have yet to define FIRM CODE and PRODUCT CODE, not even the copy protection system. Subsequently, *AxProtector* replaces the WUPI dll in the compiled application by a static implementation, and the IDs are translated into FIRM CODE and PRODUCT CODE. Optionally, an ID is able to simultaneously support *CodeMeter* and *CodeMeterAct*.

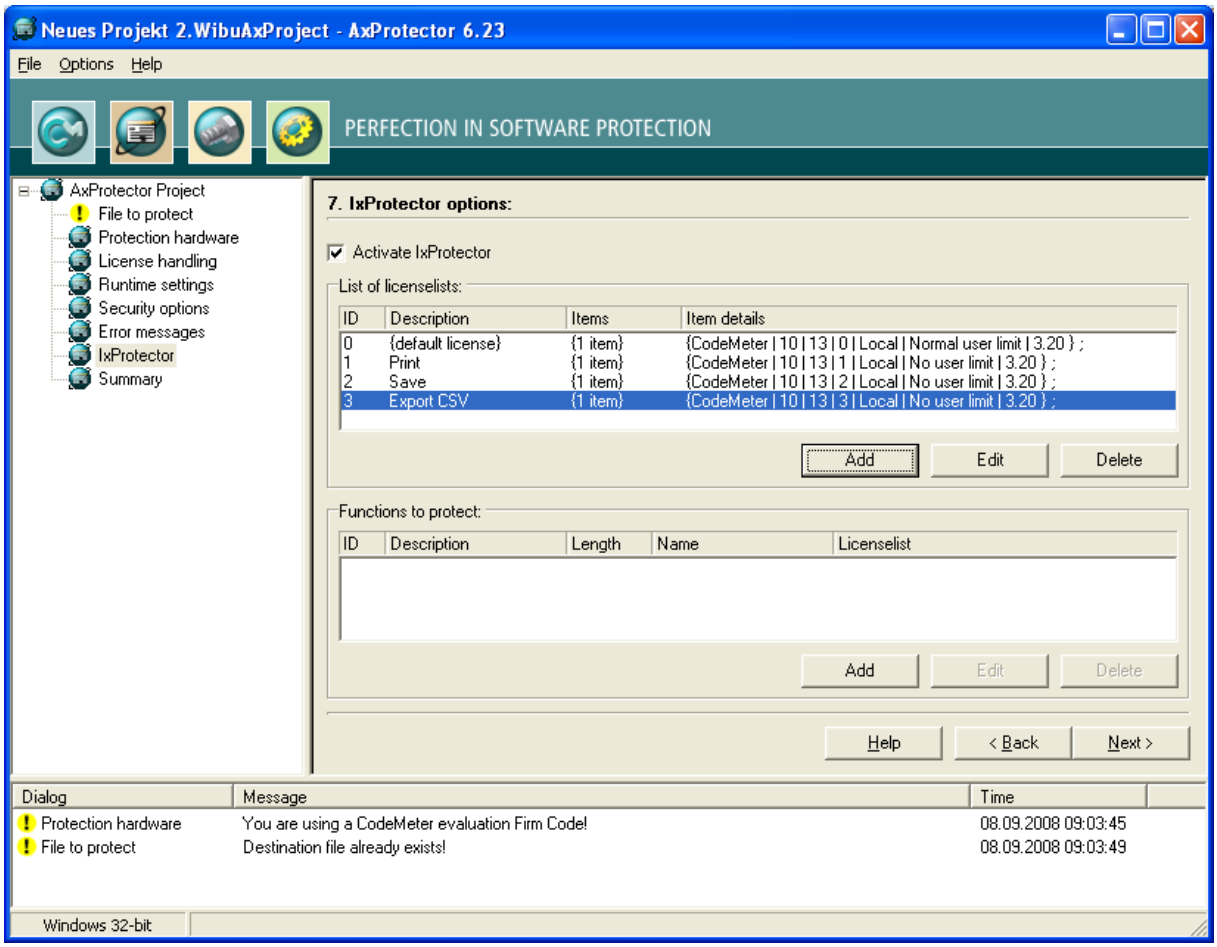


Figure 15: Supplementary definition of license details with *AxProtector*

This procedure bears many advantages:

- Security for a static library
- Subsequent selection of the copy protection system, i.e. subsequent switching between *CodeMeter* and *CodeMeterAct* without recompiling
- Separation of license details and implementation eases implementation
- Always updated mechanisms (and thus actual security) by re-encryption with *AxProtector*.

WUPI function overview:

Function	Description
WupiCheckLicenseId (LicenseId: Integer)	Allocates the license which has been defined in *AxProtector* with Id = LicenseId. Used where the definition in *AxProtector* holds *CodeMeter* and *CodeMeterAct*, then the initially defined license is queried first. Where the first is non-existent, the second license is queried. A definition may hold any number of optional licenses, even with the same copy protection system
WupiFreeLicenseId (LicenseId: Integer)	Releases the license as defined in *AxProtector* with Id = LicenseId.
WupiQueryInfoId (LicenseId: Integer, Type: Integer)	According to type, information is read from the currently allocated license entry. For example, UNIT COUNTER status, FEATURE MAP set, days remaining until EXPIRATION DATE.
WupiDecryptCodeId (FunctionId: Integer)	The function defined in *AxProtector* with Id = FunctionId is decrypted. This call has to be initiated before executing an encrypted function, otherwise the software shuts down.
WupiEncryptCodeId (FunctionId: Integer)	The function defined in *AxProtector* with Id = FunctionId is encrypted. Call this function after executing the protected function, to ensure that it gets safely stored in the memory.
WupiCheckDebuggerId (LicenseId: Integer)	This function calls the Debugger Check from within *AxProtector*. In case the Debugger Check detects a debugger, you are able (depending on option selected) to permanently lock the license defined in *AxProtector* with Id = LicenseId.
WupiDecreaseUnitCounterId (LicenseId: Integer, Units: Integer)	This function counts down the UNIT COUNTER by a favoured number for a license as defined in *AxProtector* with Id = LicenseId. You use this to integrate a pay-per-click functionality into your software. In case the license does not have a UNIT COUNTER, this function does not return an error, i.e. by programming the license you are able to define whether the licensee receives a pay-per-click or an unlimited license.
WupiGetNativeHandleId (LicenseId: Integer)	This function gets a handle to be further processed in the native API. This allows you to access extended functions of the low-level API.

Table 8: WUPI Functions Overview (excerpt)

7.3 The Native *CodeMeter* / *CodeMeterAct Core API*

Along with *AxProtector* and WUPI, WIBU-SYSTEMS provides you with a low-level API, which you can use in all other circumstances.

CodeMeter / *CodeMeterAct* provide a standard API. Only the FIRM CODE differs for both systems. *CodeMeterAct* offers additional functions to allow the integration of activation into your own software.

The API is available for the following programming languages: for the C-API a great number of interfaces for additional programming languages exist (Delphi, Fortran, Visual Basic,…).

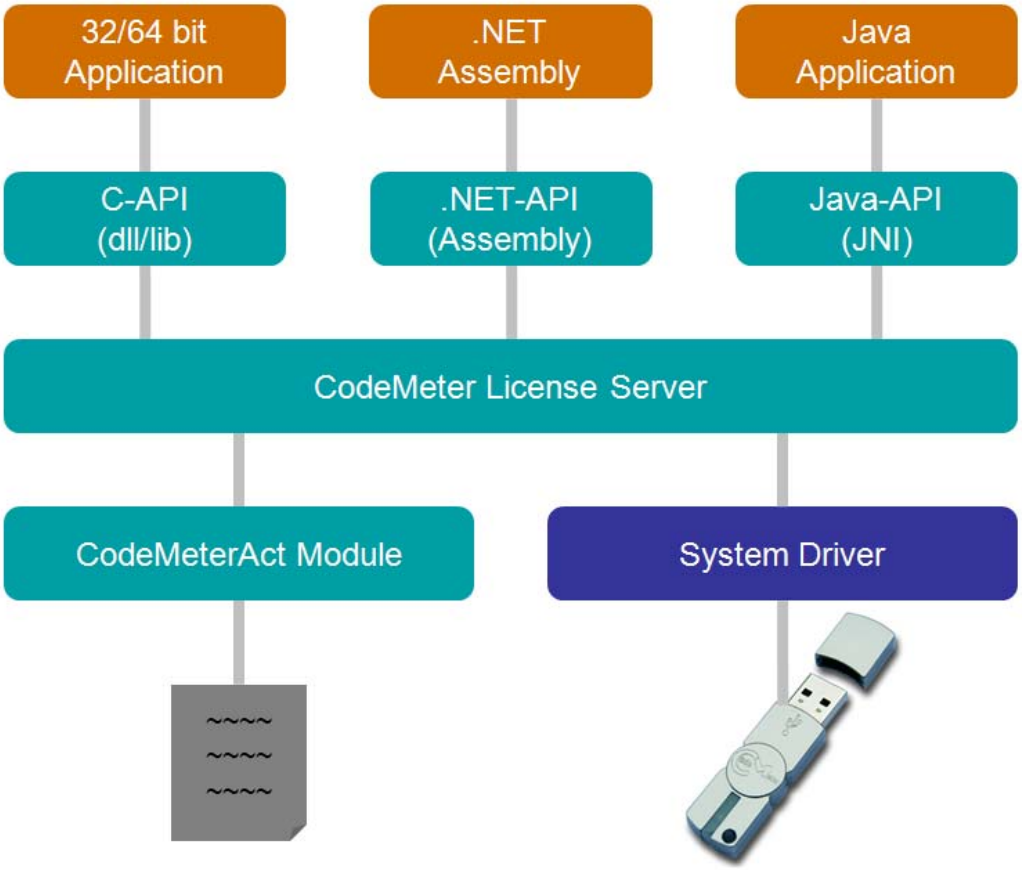


Figure 16: Available *CodeMeter* APIs

The basic functionality is provided by **CmAccess** (allocate license), **CmCrypt** (encrypt or decrypt data) and **CmRelease** (release license).

If you want to simultaneously use *CodeMeter* and *CodeMeterAct*, you can use the same function calls and concepts.

8 License Management - Backoffice Integration

8.1 The Principle – Ticket System

Integrating software protection into the software is one but fundamental aspect which strongly affects system security. At the same time, the integration of software protection into sales, production and support processes also determines whether a system is easy to operate, and thus is accepted by both customers and employees. The latter processes we summarize as Backoffice Integration (BOI).

CodeMeter License Central is a standardized ticket system at the center of this Backoffice Integration.

When you program a *CmStick*, or a *CodeMeterAct* license file for a specific product, you send a related request with an item number to *CodeMeter License Central*, and receive back a unique ticket. Since this scenario in most cases involves the selling of this item, we refer to this interface as the Sales Interface. The ticket contains the authorization to add the license to a *CmStick*, or to a *CodeMeterAct* license file.

You decide whether you instantly program the license yourself, deal with it later, or transfer the ticket to your customer. If you decide to transfer, then your customer is able to collect the licenses bought at any time, for any *CmStick*, or any *CodeMeterAct* license. We call the interface for collecting licenses: the Depot Interface.

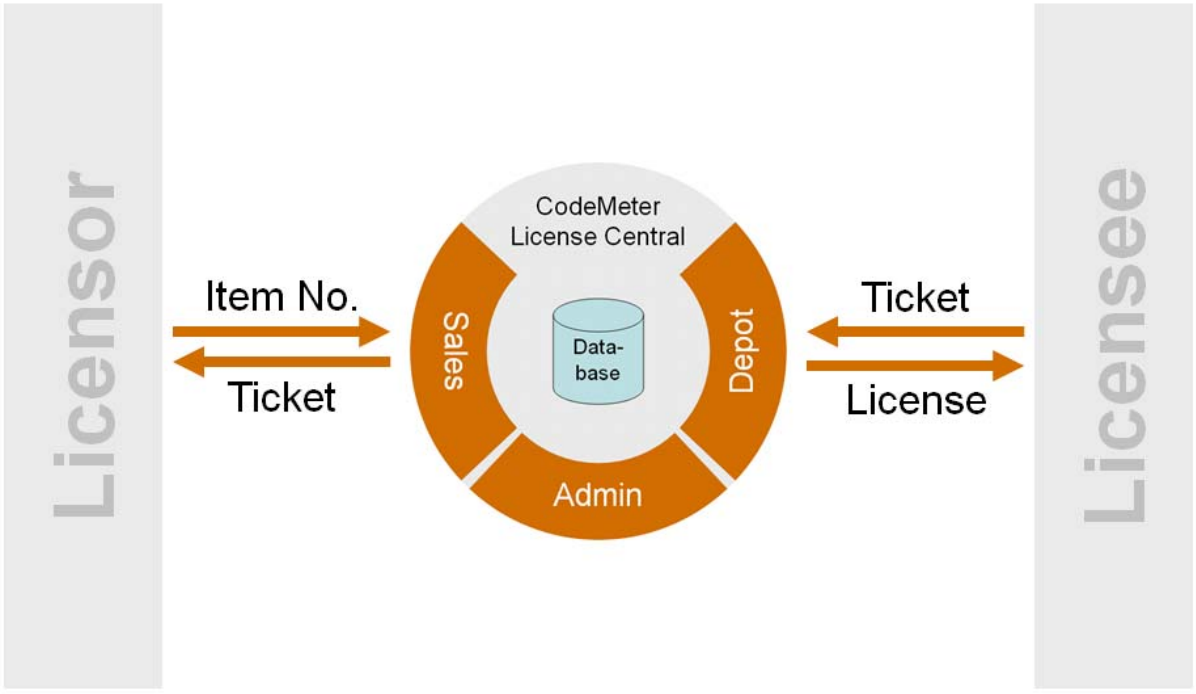


Figure 17: Collection of a License by the Licensee

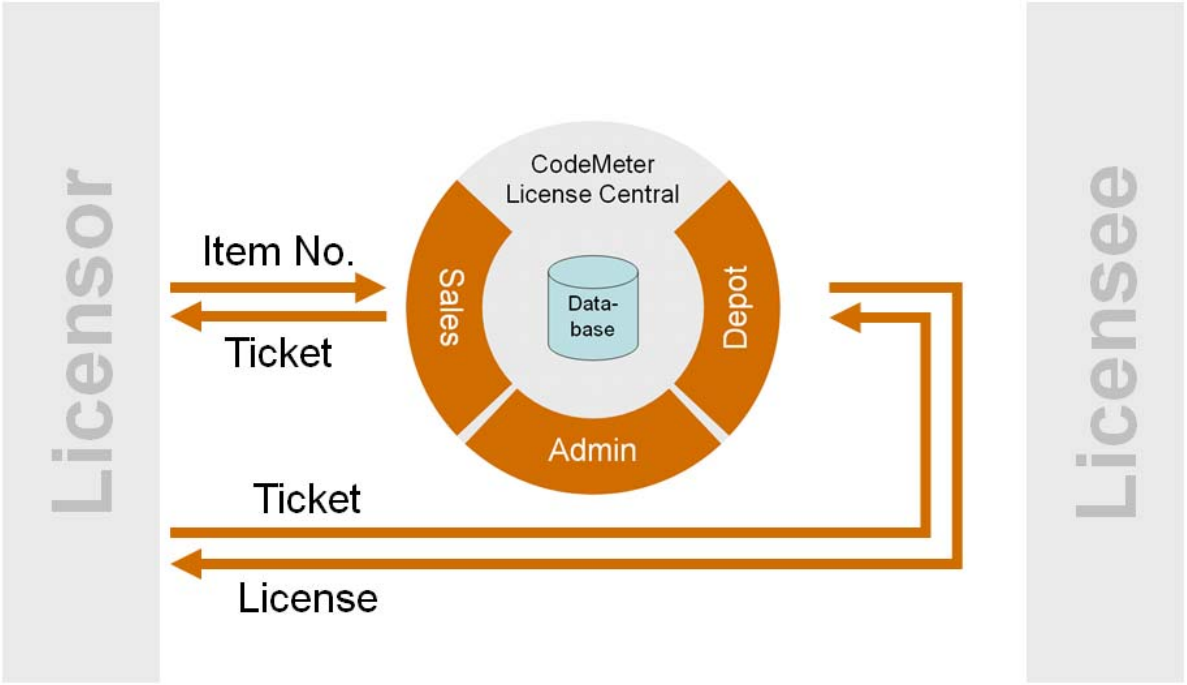


Figure 18: Collection of a license by the licensor

Next to the Depot and Sales Interface, *CodeMeter License Central* features an Admin Interface. The Admin Interface comprises functions for defining license properties (e.g. expiration time, license quantity), for managing access rights, for generating statistics and reports, and for carrying out support activities.

8.2 *CodeMeter License Central* – The Core

8.2.1 Architecture

The core of *CodeMeter License Central* consists of a database and webservices for the Sales, Depot and Admin Interface.

The webservices are cross-platform and are available in Java. A Tomcat application server is a prerequisite.

The webservices provide a SOAP based interface to *CodeMeter License Central*. The complete communications is handled by those webservices, and the webservices have a separate internal interface to the database.

Databases supported include MySQL (Windows / Linux) and MSSQL (Windows). On request, other databases can be integrated.

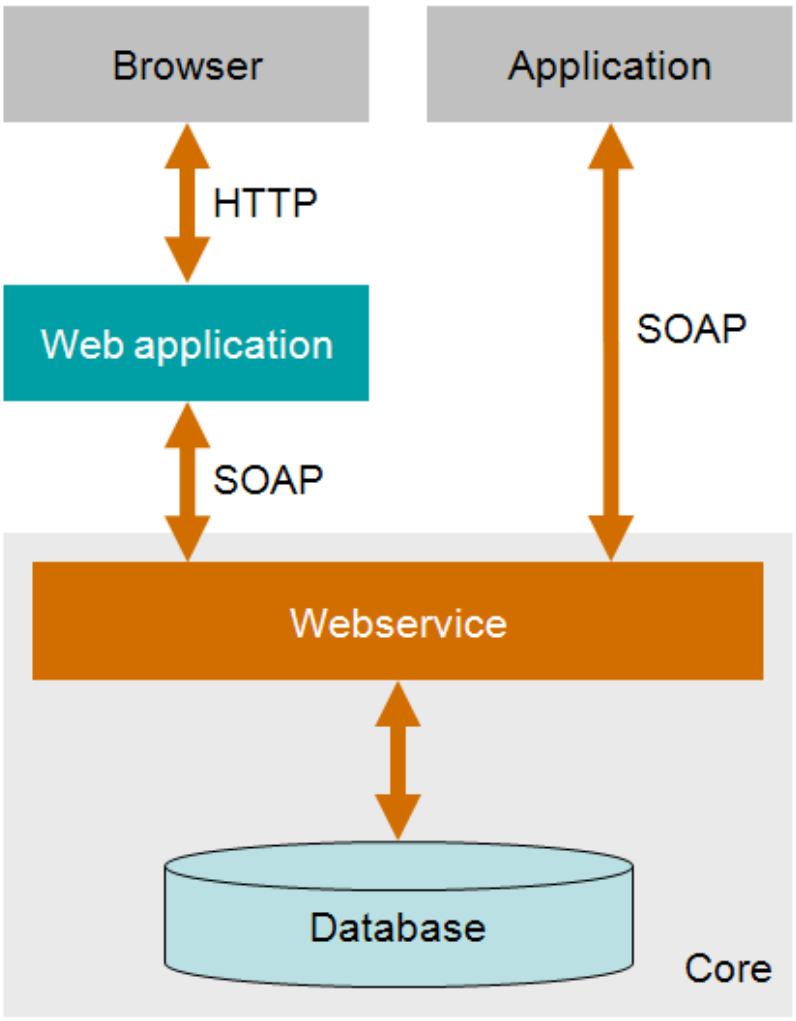


Figure 19: SOAP accesses to the *CodeMeter License Central*

SOAP accesses provide you a maximum in flexibility. You are able to access *CodeMeter License Central* from within an application. Next to the direct access, the web access is also feasible using a web browser.

Both access options are valid for all three interfaces. Also splitting is an option. For example, you are able to call the Depot Interface from within your software, while the internal interfaces, i.e. the Admin and Sales Interface are accessed via web pages.

SOAP functions are available as documented *CodeMeter License Central* API (CmLC-API) for integrating into your own applications.

8.2.2 Basic Functions

Sales Interface

The Sales Interface accepts activities. You send the item number of the product to be shipped, and optionally customer data and an order number. The Sales module returns the matching ticket.

In case of recurring activities (checkpoints, license extension), you are able to send along the original order number. In this case, the existing ticket is extended by a collection activity. This means, the user is able to extend the license with his established ticket. This saves management efforts for new tickets, and eases handling for the user. You can also extend or renew the license with the existing ticket directly via SOAP from within your application.

Depending on the item configuration, you are also able to dynamically transfer parameters on an activity. For example, you transfer the number of network licenses, or the purchaser's name written to the property: Customer Owned License Information.

Depot Interface

The Depot Interface features license collection. The collection involves the upload of a content file and the download of an update file. Optionally, after an update file is activated, a new context file may be uploaded in order to send a receipt for the license activation. Of course, this process can be achieved in one step so that the user is collecting the license only.

The Depot-Interface offers two options to collect licenses:

- **Direct** (PC with the CmStick to be programmed has Internet access.)
- **Indirect** (Activation data is transferred to another PC via file transfer.)

Next to the license collection, the Depot Interface also provides for methods for returning licenses.

Currently the feature of returning licenses is not yet implemented.

After returning the license, the user receives a new ticket. He receives it only after uploading the receipt. Using this new ticket he or she is able to transfer the license to another PC, or is able to resell it passing the ticket to the new user. If you allow the reselling of licenses, then simply activate the option: License Returning. By default, license returning – and thus reselling – is disabled.

Moreover, in the Depot Interface you are able to retrieve information on sold and activated licenses.

Depending on the product configuration, you are able to preset the copy protection system for the end user (*CodeMeter* / *CodeMeterAct*), or let the user opt for hardware-based or software-based protection.

Admin Interface

The Admin Interface consists of the following parts; license configuration, evaluation, support, and user management.

In license configuration you are able to manage license properties and the related item numbers. Here you individually define for each license which parameters are preset, and which are dynamically transferred to the Sales Interface.

In the statistics module you are able to evaluate data from *CodeMeter License Central*; for example licenses on *CmSticks* per customer.

For closing open processes (e.g. receipt not uploaded), the release of further activations, and the editing of blacklist entries; you use the support module.

User management provides you with the option to configure the "access privileges" to *CodeMeter License Central*. Those include user name, password, IP range, and *CmStick* signature. For example, you can set it up so that a sales partner with changing IP addresses has to authenticate using a *CmStick*, while a Sales Partner with a web portal must log on using an authorized IP address.

8.3 CodeMeter License Central Desktop

With *CodeMeter License Central Desktop* we provide you a complete package containing not only the core but also complete web applications allowing you to access all functions of *CodeMeter License Central*.

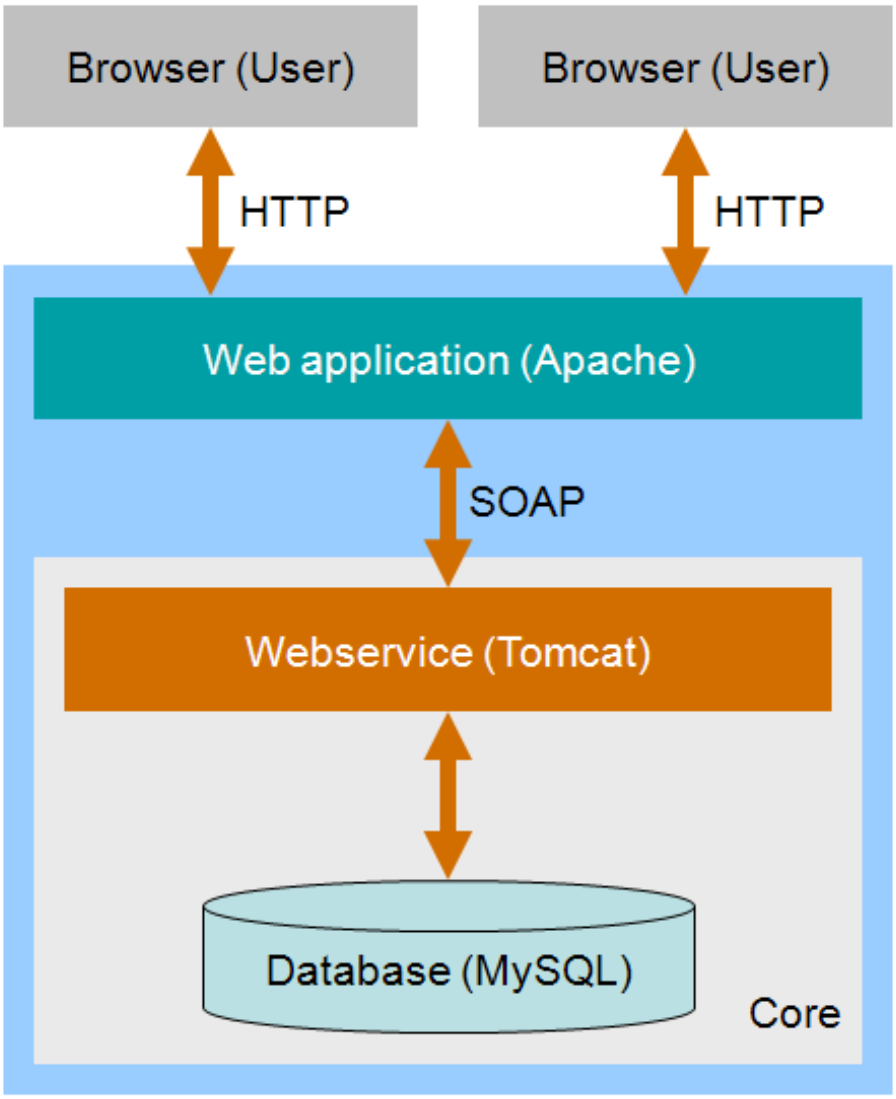


Figure 20: *CodeMeter License Central Desktop* Architecture

CodeMeter License Central Desktop, as a completed package, is available for a Linux VmWare Image provided by WIBU-SYSTEMS. The package includes a MySQL database, a Tomcat application server and an Apache web server.

Access to *CodeMeter License Central Desktop* is provided by web applications running in the same VmWare image on the Apache web server. Simultaneous multi-user access via web applications is feasible. A TCP/IP connection to the VmWare with *CodeMeter License Central Desktop* is the only thing required.

CodeMeter License Central Desktop features the following modules:

- **Sales – Manual** (manual input of orders)
- **Depot – Get** (download of license with ticket)
- **Depot – Return** (returning of license)
- **Admin – Statistics** (customers, orders, licenses)

- **Admin – Support** (locking, unlocking, …)
- **Admin – Products** (administration of *CodeMeter* properties of items)
- **Admin – User** (management of users with privileges to use *CodeMeter License Central Desktop*)

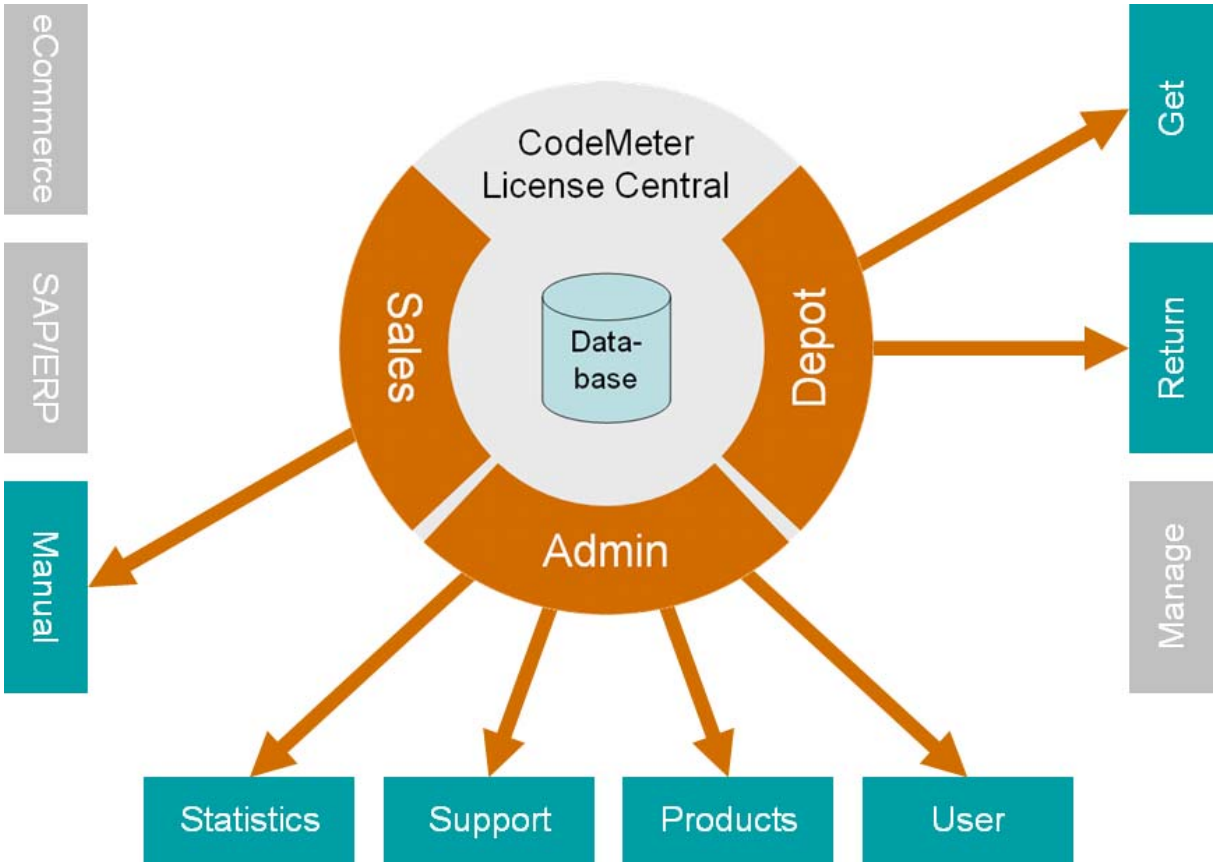


Figure 21: *CodeMeter License Central Desktop* Modules

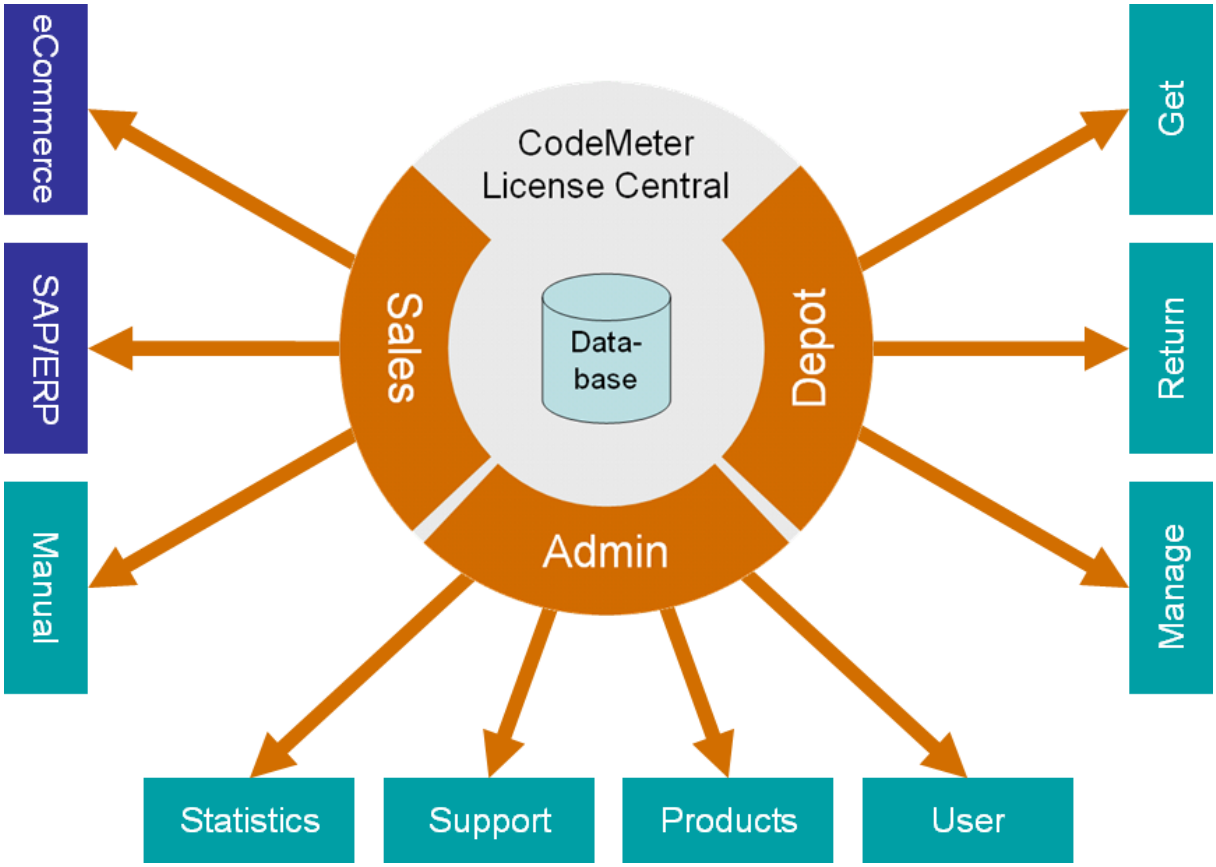


Figure 22: *CodeMeter License Central Enterprise* Modules

8.4 CodeMeter License Central Enterprise

8.4.1 Differences to the Desktop Version

In contrast to *CodeMeter License Central Desktop* where you receive a complete package, *CodeMeter License Central Enterprise* can be customized to meet your requirements.

CodeMeter License Central Enterprise features additional modules allowing for the automatic integration into sales processes by Connectors. In *CodeMeter License Central Enterprise*, the generation of tickets is replaced by a separate rule-based system. Also a distributed installation (web services, web applications and database) on several computers is feasible.

The following table compares features between *CodeMeter License Central Enterprise* and *CodeMeter License Central Desktop*.

Property	Desktop	Enterprise
Core - Database	MySQL	MySQL, MSSQL, additional on request
Core - Java web services	X	X
Core – Customizable ticket generation		X
Core – Distributed installation		X
Sales - Manual interface (web interface)	X	X

Sales - ERP-Connector		X
Sales - eCommerce Connector		X
Admin – Statistics (web interface)	X	X
Admin - Support (web interface)	X	X
Admin – Product management (web interface)	X	X
Admin – User administration (web interface)	X	X
Depot – Download with ticket (web interface)	X	X
Depot – License return (web interface)	X	X
Depot - License management (web interface)		X
Depot – Download with user account (web interface)		X

Table 9: Feature Scope *CodeMeter License Central*

8.4.2 Sales Connector

In order to allow an automatic integration into the sales processes after the successful sales activity; the ERP system or online shop will have to initiate a process.

eCommerce Connector

Many online shops, such as, Cleverbridge, ShareIt, Digital River, Element5, etc., provide SOAP access to a license generator. Since each shop speaks a different 'dialect', you will require an eCommerce Connector, which transfers online shop data into interpretable data for *CodeMeter License Central*. The eCommerce Connector then sends this data to the Sales Interface, and gets the ticket in return. Depending on the online shop, this ticket may need to be processed further, and then returned either as license string, or as collecting link. This transformation is provided by the eCommerce Connector.

The ticket display, for example as a URL encoded collection link, is configurable in the online shops. In most cases, the ticket can be integrated into the order confirmation page and the order confirmation mail.

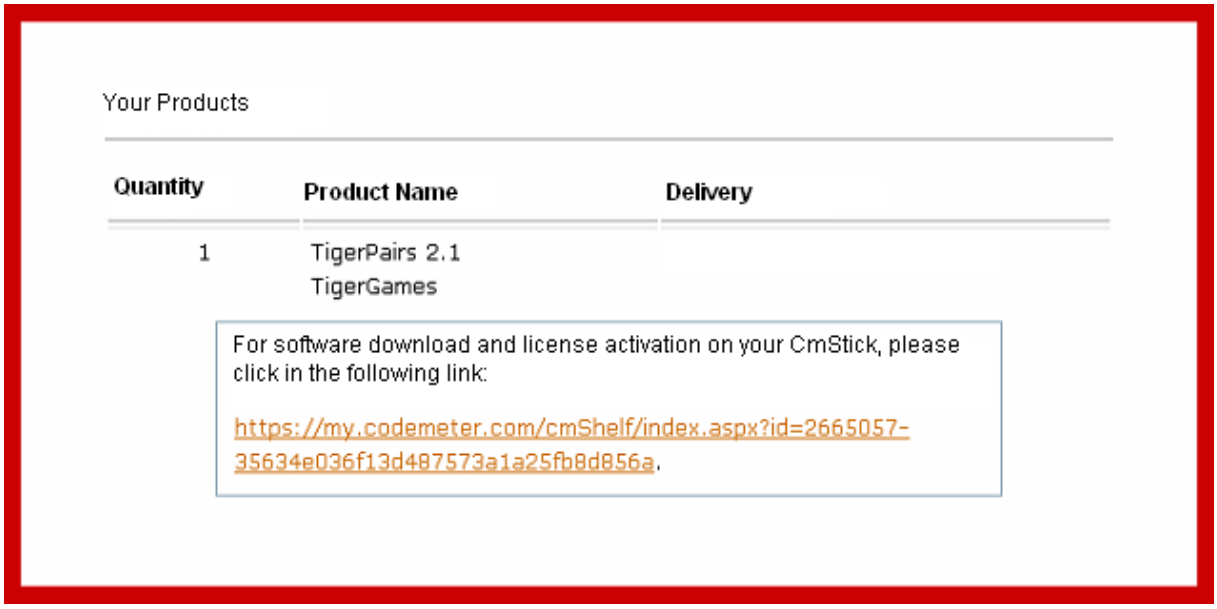


Figure 23: Ticket Display as Collection Link in Cleverbridge Online Shop

The eCommerce Connector authenticates with *CodeMeter License Central*. On the other side, the eCommerce Connector has to ensure that the request originates with the correct online shop. This can be provided by IP range check, or by individual login.

The eCommerce Connector has to be customized for each online shop.

ERP Connector

The ERP Connector provides for the integration into an ERP System, for example SAP.

ERP system customizing is rather marginal:

1) After order recording the ERP Connector is addressed, for example, realized as Dynamic Link Library (DLL) directly accessed from within the ERP system. The ERP Connector sends the order data to CodeMeter License Central and receives the ticket back. The ERP Connector passes the ticket on to the ERP system.

2) The ticket is to be saved in the ERP system, and to be printed on the order. Many ERP systems provide user-defined fields.

In case this ERP system extension is not feasible, you are able to work with a shadow database. That means, the order data is exported from the ERP system to this shadow database, and the ERP Connector is alternatively addressed from there.

8.4.3 Collecting by the Licensee

In case the licensee is to directly collect the licenses, an access to *CodeMeter License Central* is required. Depending on the access designed – either directly via SOAP from within the application, or via a web page – please set the web server, or web server and application server into the DMZ (inside the firewall).

In this case, for security reasons it is recommended that you distribute the installation on several computers, and to locate the remaining modules (database and eventually the application server) behind an inner firewall.

The web pages the customer sees when he is collecting his licenses is totally customizable and can follow your corporate design.

In *CodeMeter License Central Enterprise* Edition you are able to activate "user management", e.g. the end user is a large company, and several staff members need to collect the licenses. In this case the user's "manager" is able to review information on which licenses have been collected by which staff member. In this case, the user "manager" identifies with his account but not with his ticket number.

8.5 Additional Modules

By using additional modules the *CodeMeter License Central* functionalities are extendable, and provide interfaces to other systems.

8.5.1 One2One Marketing Modul (OZOMM)

The One2One Marketing Module can be used in cases of time-limited licenses. It is available as an additional module for *CodeMeter License Central Enterprise*, and provides an interface to the user's CRM system.

On trying to extend a license the SOAP request is not send to the Depot Interface but to OZOMM.

- OZOMM checks the ticket number to see if the customer is eligible for a license extension
- OZOMM extends the license via the Sales Interface
- OZOMM collects the license via the Depot Interface
- OZOMM sends additional marketing material to the customer (upgrade offers)

For offers OZOMM provides an interface to the software developer's CRM system. The check for eligibilty of the customer to extend the license is provided by an interface to the developer's CRM or ERP system.

Connectors to the developer's ERP or CRM system are individually developed for the related systems. These Connectors are able to transfer data online from related systems, or regularly import data into the developer's own databases.

9 Contact

Headquarter | WIBU-SYSTEMS AG

Rueppurrer Strasse 52-54 | 76137 Karlsruhe, Germany

Tel.: +49-721-93172-0 | Fax: +49-721-93172-22

E-Mail: info@wibu.de | Web: www.wibu.de

WIBU-SYSTEMS USA Inc.

110 W Dayton Street, Suite 204 | Edmonds, WA 98020-7245, USA

Tel: +1.425.775.6900 | Tel: +1-800-6-GO-WIBU (+1-800-646-9428) | Fax: +1-206-237-2644

E-Mail: info@wibu.us | Web: www.wibu.us

WIBU-SYSTEMS (Shanghai) Co. Ltd.

Room 1602, KIC66 Jin Chuang Road, Yang Pu District | 20 200433 Shanghai, VR China

Tel: +86-21-55661790 | Fax: +86-21-55661780

E-Mail: info@wibu.com.cn | Web: www.wibu.com.cn

WIBU-SYSTEMS BV

Adam Smithstraat 33 | 7559 SW Hengelo, The Netherlands

Tel :+31 (0)74 75 01 495| Fax: +31 (0)74 75 01 496

E-Mail: sales@wibu-systems.nl | Web: www.wibu-systems.nl

WIBU-SYSTEMS LTD

The Mansion, Bletchley Park, Bletchley

Milton Keynes MK3 6DS, United Kingdom

Tel: +44 (0)20 314 747 27| Fax: +44 (0)20 314 747 28

E-Mail: info@wibu.co.uk | Web: www.wibu.co.uk

WIBU-SYSTEMS NV

Drie Eikenstraat 661 | B-2650 Edegem (Antwerpen), Belgium

Tel.: +32 (0)3 400 03 14| Fax: +32 (0)3 400 28 40

E-Mail: sales@wibu.be | Web: www.wibu.be

WIBU-SYSTEMS Iberia

C. Josep M. Gironella, 1-3 | 17003 GIRONA, Spain

Tel: +34 (0) 91 414 8768 | Fax: +34 (0) 91 414 8769

E-Mail: info@wibu.es | Web: www.wibu.es