THE WIBU-MAGAZINE

# KEYnote51

SPRING/SUMMER 2026

## Cost Optimization Through CodeMeter for Licensing

**Highlights**

- Software Subscriptions in Industrial Environments
- Stateless Apps, Stateful Licenses: A Strategic Guide to Containerized Protection
- CodeMeter 9.00 – Architectural Transformation with Guaranteed Compatibility

WIBU SYSTEMS

# Content



EDITORIAL

# When Technology Becomes a Long-Term Commitment

This issue of our KEYnote magazine marks a moment of continuity as much as of change. Since our founding in Germany in 1989, Wibu-Systems has grown organically, opening offices where technology, industry, and responsibility intersect in a meaningful way. Belgium followed in 1995, the United States in 2001, China and the Netherlands in 2005, the UK in 2006, France in 2013, Japan in 2018, Sweden in 2021, and Korea in 2024.

In April 2026, we have formally added another point to this map with the establishment of Wibu-Systems India Pvt. Ltd., ahead of its official inauguration later this year.

This decision was not driven by a desire to be everywhere, but from the conviction to be **where it matters.** India has reached a level of industrial maturity where software is no longer auxiliary, but central to value creation, whether in manufacturing, automation, mobility, or digital infrastructure. Indian companies are building their own digital IP, exporting technology, and ensuring secure, compliant products in global markets.

At the same time, we see a shared understanding emerging. *"Strong technology ecosystems grow when sovereignty is respected on both sides, when innovation can scale internationally without surrendering ownership, control, or trust,"* and this principle has guided our engagement with India from the start.

What we gain is proximity: to engineering excellence, to customers who design software for demanding real-world environments, and to partners who expect protection, licensing, and compliance built in. What India gains is a long-term technology partner that invests locally, transfers know-how, and supports sustainable product businesses rather than short-term returns.

We believe in trade between sovereign states built on fairness, responsibility, and mutual benefit – principles that shaped industrial collaboration long before they became fashionable slogans. Opening our office in India is an expression of that belief, and of our confidence in this country as a place where secure software and trusted digital value will continue to grow.

Best regards,
Oliver Winzenried
CEO

# From CRA Requirements to Lifecycle Control

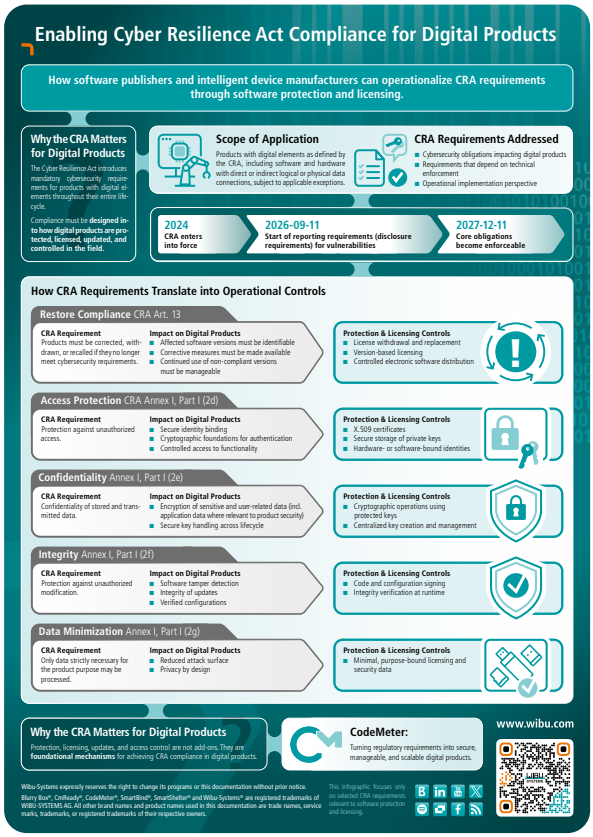
The Cyber Resilience Act (CRA) introduces binding cybersecurity requirements for products with digital elements across their entire lifecycle. Addressing these requirements in practice demands technical mechanisms that translate regulatory provisions into operations, enforceable control and maintain their effectiveness after deployment.

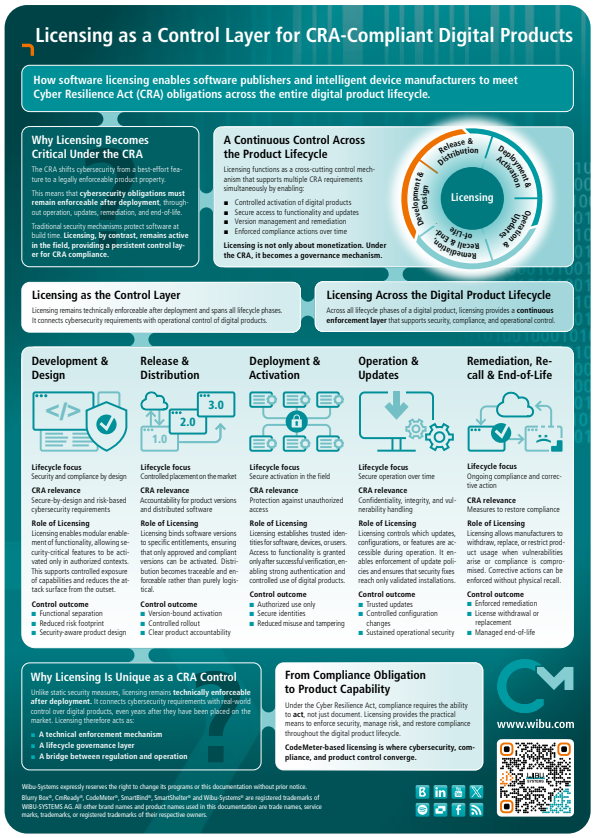
The CRA establishes a comprehensive framework of cybersecurity requirements for products with digital elements, addressing areas such as access protection, data confidentiality and integrity, data minimization, and the ability to restore compliance when vulnerabilities emerge. The regulation follows a risk-based approach and deliberately avoids prescribing specific technical implementations. This places the responsibility on manufacturers to translate regulatory expectations into practical, product-level controls.

The **first infographic** highlights how selected CRA provisions can be addressed through technical protection and licensing mechanisms that support version identification, access restrictions, update handling, and corrective measures. The emphasis is on ensuring that manufacturers retain the technical ability to manage product states, react to vulnerabilities, and maintain compliance where required. In this context, software protection and licensing provide a structured way to put CRA expectations into practice.

However, compliance does not end once a product is released. CRA obligations persist throughout operation, maintenance, and end-of-life. The **second infographic** illustrates how licensing can function as a continuous control layer across the digital product lifecycle, supporting activation, controlled use of configurations and features, remediation processes, and secure handling of products and data at end-of-life.

Together, the two infographics show a coherent path: from interpreting CRA provisions as technical requirements, to sustaining operational control over time.



# License Transfer – Trust or Control?

In general, there are three strategies for storing licenses on the user side: in separate hardware (a dongle), bound to a specific computer, or user-bound in the cloud. This article examines what happens when a user changes computers.

There may be various reasons for this. Some users operate multiple computers at different locations, while others receive new machines with improved performance. For the actual process, it makes little difference whether the software is consumer-oriented or business software. What matters far more is whether the user is permanently, or at least regularly, online. A similar situation occurs in business environments when a user changes roles and a new user takes over the license.

## Simple and Fast: The Dongle

With a dongle such as a CmDongle, the process is straightforward. The dongle is handed over like a relay baton, from computer to computer or from user to user: unplug, plug in, and continue working.

Users who would prefer to install and use the software simultaneously on multiple devices typically raise two concerns:

*"I can't work if I forget the dongle."*

Let's consider three application scenarios:

- **The user works on a mobile computer:** In this case, the dongle is typically connected directly to the computer. CodeMeter provides an especially compact USB dongle designed specifically for this purpose.
- **The user accesses their computer via Remote Desktop:** Here, the dongle remains plugged into the remote office computer where the software is running.
- **The user works on a computer at the destination location:** In this scenario, planning is required, including the installation of the appropriate version of the software. This planning should also take the dongle into account.

*"My colleague, with whom I share the license, is at another location. I can't mail the dongle back and forth every time."*

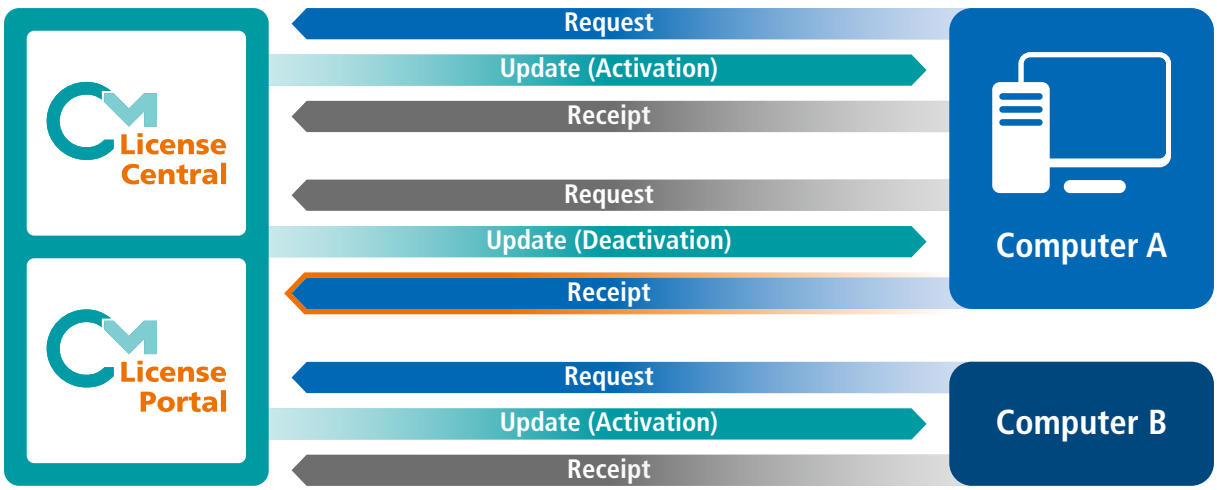



Figure 1: Receipt Required

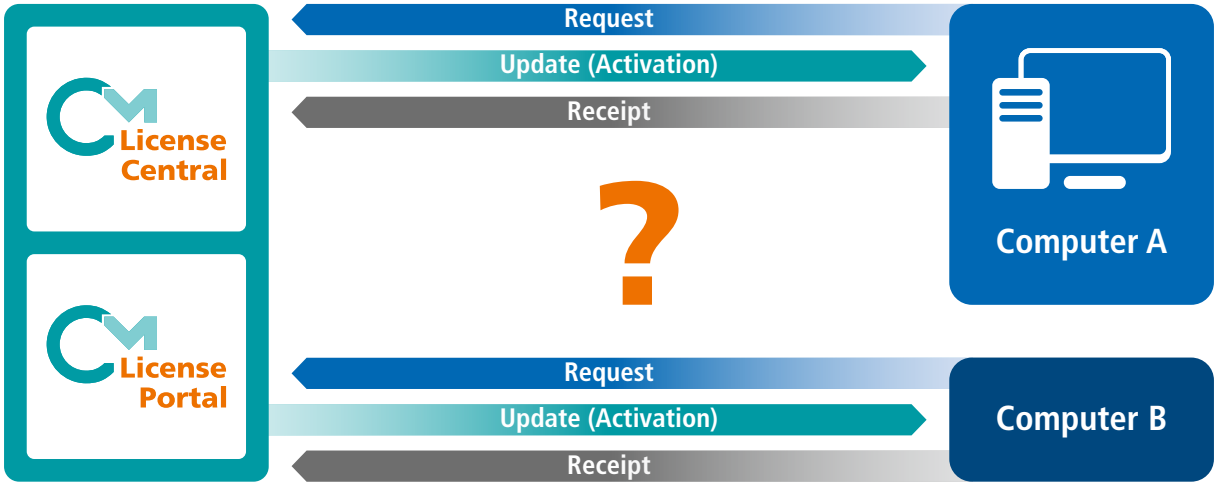



Figure 2: Reactivation Allowed

This is precisely what a single-user license is designed to prevent. For such cases, floating network licenses are available, which can be shared via a local license server (VPN) or a cloud-based license server.

## Designed for Mobility: License in the Cloud

With a purely cloud-based license such as a CmCloudContainer, licenses are permanently stored and used in the cloud. CodeMeter Cloud provides Personal and Enterprise CmCloudContainers.

A Personal CmCloudContainer is tied to an individual user and contains their personal single-user licenses. By logging into the software vendor's License Portal, the user can connect the container to any computer. They can then use their licenses interchangeably from different machines. Simultaneous use is monitored in the cloud so that a license can only be used by one computer at a time.

An Enterprise CmCloudContainer can be shared directly or indirectly among multiple users. This allows floating network licenses to be implemented in the cloud. In this case, concurrent usage is also counted in the cloud.

In principle, this is an optimal solution. However, users raise the concern:
*"This doesn't work if I'm offline or if my Internet connection is unstable."*

Personal CmCloudContainers currently require a continuous online connection. For Enterprise CmCloudContainers, license borrowing provides a ready-made solution. A selected number of licenses can be temporarily borrowed to the user's computer and used offline during that period. During the borrowing period, the license remains reserved on the cloud server and cannot be used elsewhere. The borrowing duration can be defined by the software vendor: the longer the period, the less frequently the user needs to be online; the shorter the period, the faster the user can switch to another machine.

## Offline Use: License on the Computer

The most cost-effective option is a local license file such as a CmActLicense. This eliminates the cost of external hardware (dongle) or cloud server infrastructure. Only a server for license activation is required. Offline usage is also possible, including file-based license transfers for fully air-gapped systems.

But how does license transfer work in this case? Three scenarios must be considered:

1. The old computer is available.
2. The old computer is currently unavailable but may become available later.
3. The old computer is no longer available (for example, it has been decommissioned).

With CodeMeter License Central and CodeMeter License Portal, the software vendor can select or combine various options.

## Receipt Required

The default option requires a deactivation receipt from the old computer before the license can be activated on the new one. This setting prevents any license misuse because there is never a duplicate copy of the license.

However, this approach does not address scenarios (2) and (3) and can be cumbersome for fully offline systems. The user transfers the deactivation update from an online computer to the offline machine, retrieves the receipt confirming execution from the offline machine, and uploads it via WebDepot or CodeMeter License Portal using the online computer.

## Reactivation Allowed

In addition to the receipt requirement, the software vendor may allow reactivations (license recovery). In this case, within a defined limit (for example, once initially and once per year), the user may activate the license on a new computer without returning it from the old one.

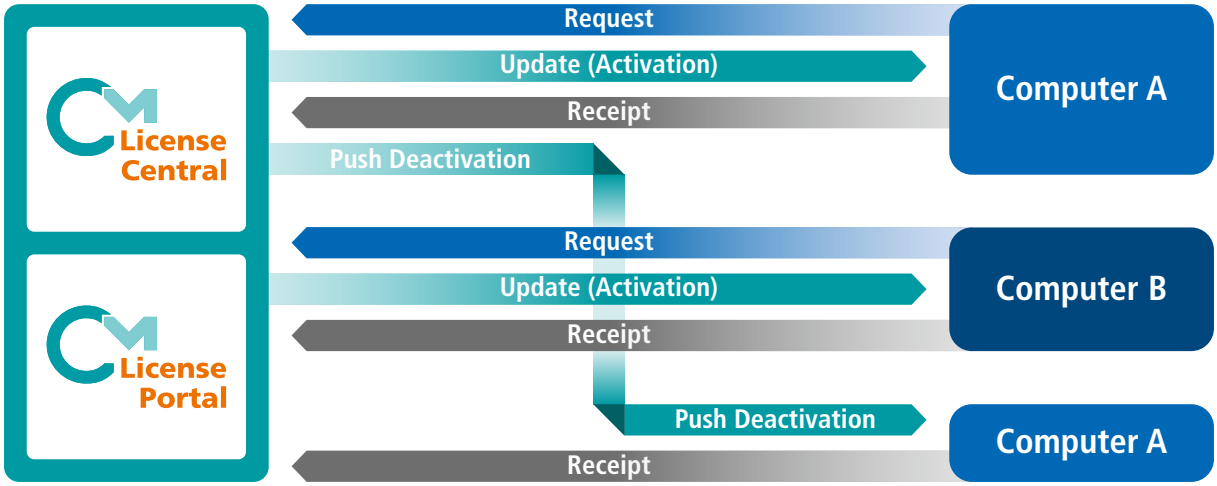



Figure 3: Push Update

This covers the third scenario, but creates a gray area if the old computer still exists and is in use. Once the permitted number of reactivations has been reached, the software vendor can manually authorize additional reactivations for specific licenses.

Possible countermeasures to prevent permanent duplicates in scenarios (1) and (2) include sanctions such as withdrawing the license or blocking the entire computer if it reappears. Continued use on the old computer would then require manual intervention by the software vendor.

## Push Deactivation

Starting with version 25.09, a third option is available in CodeMeter License Portal: licenses can now be deactivated on the old computer via push update. A push update is generated on the server without requiring prior contact with the old computer. The update is automatically confirmed in the background, allowing the license to be immediately reactivated.

As long as no receipt confirming execution of the push update is received, CodeMeter License Central will repeatedly deliver the push update whenever any programming action occurs on that computer (more precisely, within that CmActLicense). This means that at the next opportunity, the license is removed from the old computer, without penalizing the user for further use of that machine.

This option covers all three scenarios. The license is activated on the new computer and removed from the old one, when possible. License transfers are logged in CodeMeter License Central, enabling reports that show how often users have transferred licenses without subsequent removal from the old computer. Individual rules can be defined to specify when further transfers require explicit approval from the software vendor.

## Online-Update

Particularly in the third scenario, where the old computer is no longer available, the software vendor must trust the user that the license is not being used twice. Unfortunately, it is impossible to distinguish whether the old computer truly no longer exists.

An additional safeguard is to integrate an online update mechanism into the software, for example through a Software Activation Wizard. This component periodically (for example daily or weekly) contacts CodeMeter License Portal and checks for pending updates for the CmContainer. If updates exist, they are downloaded, applied, and a receipt is returned.

In addition to deactivated licenses, subscription renewals or automatic updates to new versions or features can also be handled in this way. This mechanism is available not only for CmActLicenses but also for CmDongles and CmCloudContainers.

## Time-Limited Licenses

Another option is to issue short-term licenses, such as those limited to 30 days. The license is then automatically renewed in CodeMeter License Portal. If the customer is at least occasionally online, the license in the new CmContainer can be automatically renewed. A renewal attempt on the old computer deletes the license there.

If the old computer remains fully offline and receives no license updates, the license will automatically expire. Push updates can simplify the renewal process on the old computer to a monthly download of the extension.

CodeMeter License Central and CodeMeter License Portal offer multiple options for managing license transfers. Particularly in cases where the original license has been lost, the software vendor can decide whether to approve each case manually or offer the user a limited self-service recovery option. Through regular online synchronization and time-limited licensing, potential misuse of self-service recovery can be significantly reduced.

# How to Get Your CodeMeter SDK Licenses

The CodeMeter SDK uses CodeMeter for its own licensing, making it an ideal reference case. For software vendors, this means you experience the same user journey that your customers will encounter. A central component in this process is CodeMeter License Portal. Software vendors can customize and operate a License Portal for their own users. The CodeMeter License Portal where software vendors receive and activate their developer licenses for the CodeMeter SDK is called CodeMeter Developer Portal.

## What Types of Licenses Are Available?

The CodeMeter SDK includes tools for software protection and license generation. Both categories of tools require licenses, and both rely on a shared key store.

### Software Protection

These tools, named AxProtectors, are bundled in CodeMeter Protection Suite and encrypt software before it is delivered from the software vendor to the end user. They are available for various programming languages and operating systems and automatically apply the most appropriate protection methods for the given environment. Depending on configuration, AxProtector operate on source code, intermediate code, or compiled machine code.

All variants of AxProtector share a common structure: there is a base version and several extensions. The base version enables complete protection of an application. Examples of extensions include:

- **Modular Licensing**, which allows individual functions within an application to be licensed separately.
- **Unbound Mode**, which enables protection against reverse engineering without delivering a license.

All AxProtector licenses are available as **subscriptions** and include access to the current version as well as downgrade rights to all versions from the previous two years.

### License Creation

To generate licenses, a license to create licenses is always required. This license is available in two options:

- **Unlimited License:** An annual license, priced according to the value of the protected and licensed software.
- **License per CmContainer:** A volume-based license that measures the number of generated CodeMeter containers.

Simple tools such as the command-line tool **CmBoxPgm** and the graphical **CodeMeter License Editor** do not require any additional license beyond this container-generation license.

**CodeMeter License Central** and **CodeMeter License Portal**, however, require an additional license depending on the selected configuration level and deployment volume. These licenses, like AxProtector licenses, are available as **subscriptions**.

## What Is the Key Store?

In addition to the required licenses, the software vendor needs a key store. CodeMeter security is based on encrypting the software and separating the cryptographic key from the executable file. AxProtector therefore requires a key when encrypting the software, and the tools used to generate licenses require that exact same key.

For this reason, the software vendor needs a key store in addition to the tool licenses. The key store contains multiple master keys from which the operational keys are derived. Beyond encrypting the software, these keys are also used to digitally sign software and licenses.

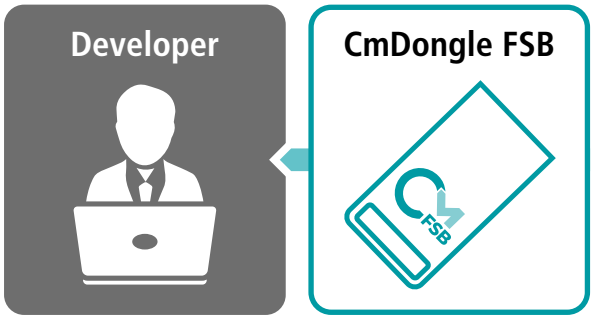


Figure 1: Licenses for Individual Developers

## Which CmContainers Are Supported?

CodeMeter offers the option to store licenses in **CmActLicenses**, **CmDongles**, and **CmCloudContainers**. In addition to licenses, the key store is also managed within the CmContainer.

Due to the differing security levels of the three CmContainer types, and the extremely high protection requirements of the key store, Wibu-Systems has decided that CodeMeter SDK licenses and the key store may only be stored in CmDongles and CmCloudContainers. These specific containers are referred to as **CmDongle FSB** and **CmCloud FSB**.

Depending on the use case, a software vendor may receive one or multiple CmDongle FSBs and/or CmCloud FSBs, and both types can be combined.

In most cases, many developers and systems perform encryption, while only a few (or a single dedicated system) generate licenses. It is possible to obtain an FSB that can both protect software and generate licenses. This is referred to as a "standard" FSB and is typically used for the system that generates licenses. Additionally, it is possible to obtain an FSB that can only perform encryption but cannot generate licenses. This is referred to as an **Encryption-Only FSB**.

## Licenses for Individual Developers

The simplest option, particularly for smaller projects, is to assign a personal FSB to a developer. In this case, either a CmCloud FSB or a CmDongle FSB can be used. For a CmDongle FSB, the key store is either delivered directly with the dongle or transferred manually via an update file for technical reasons. For a CmCloud FSB, the key store is activated in the CmCloud FSB through CodeMeter Developer Portal. In this scenario, a personal CmCloudContainer tied to the individual user is automatically created in the background during activation. If the developer is authorized to generate licenses, the corresponding license is also activated via CodeMeter Developer Portal.

For AxProtectors, User and Enterprise licenses are available. A User license is limited to a single user and includes a 30-minute reuse restriction, meaning it can only be used again on the same system within 30 minutes after its last use. For scenarios in which AxProtector must run on different systems, an Enterprise license is required.

## Company License Server

If multiple developers or automated build systems need to encrypt software, the choice between a CmDongle FSB and a CmCloud FSB depends on whether the license server should operate offline.

The CmCloud FSB is typically created by an administrator at the software vendor for a technical user. The credential file associated with the FSB is installed on the license server. Through an access control list (ACL), the administrator defines which developers or systems may use the licenses and key store.

In most cases, Encryption-Only FSBs are used alongside Enterprise AxProtector licenses. Enterprise licenses can be used on up to 100 computers simultaneously and come with no reuse restriction.

## Cloud-Based Build Pipelines

If an AxProtector is to be used within a cloud-based build pipeline, for example in Microsoft Azure, a CmCloud FSB is required. Since this FSB is not assigned to an individual developer, the administrator creates a technical user and the corresponding CmCloud FSB, similar to the license server scenario. The credential file is then downloaded and integrated into the build process.

Within the CmCloud FSB, the key store and an Enterprise version of the required AxProtector are activated. Unlike a centralized license server, the credential file is integrated into each build process (typically within containers) rather than managed centrally.

## Self-Hosted CodeMeter License Central

When operating CodeMeter License Central and CodeMeter License Portal either on-premises or in the cloud, the general procedure is similar to the license server scenario. Vendors can

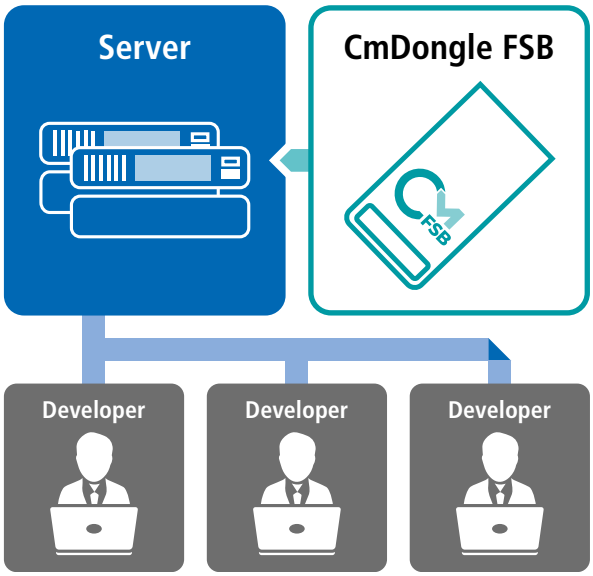


Figure 2: Internal License Server

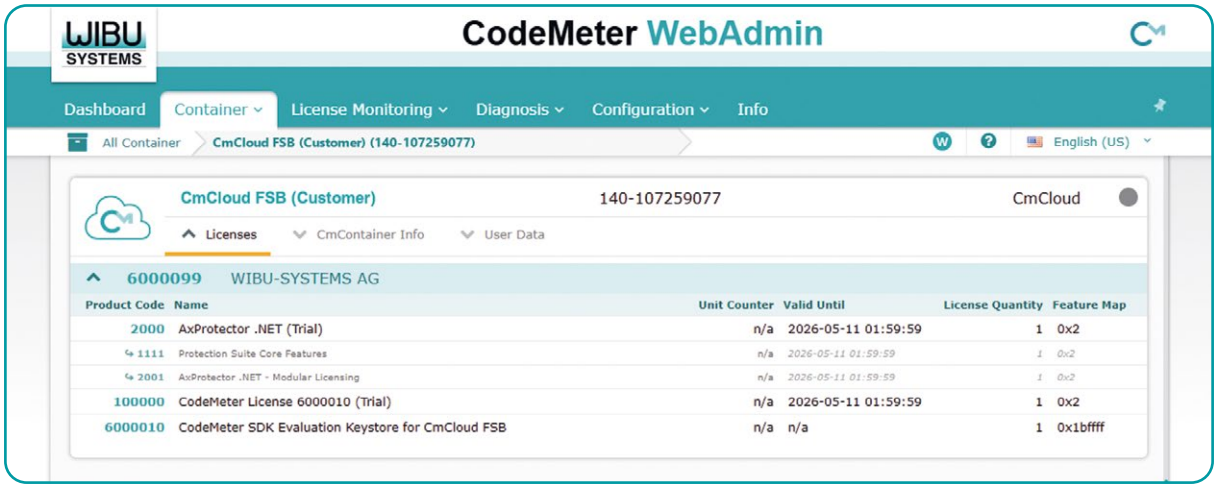



Figure 3: Sample of a CmCloud FSB

choose between a CmDongle FSB (local only) and a CmCloud FSB (local or cloud).

In the case of a CmCloud FSB, a technical user is used, and the credential file is integrated directly into CodeMeter License Central and CodeMeter License Portal. A CmDongle FSB is either directly connected or made available exclusively to a specific machine or IP address via a network license server.

Within the FSB, the key store, the license for generating licenses, the license for CodeMeter License Central, and optionally, the license for CodeMeter License Portal must be activated.

## Hosted CodeMeter License Central

When CodeMeter License Central, and optionally CodeMeter License Portal, are operated by Wibu Operating Services (WOPS), licenses, the key store, and the license-generation licenses are managed via CodeMeter Developer Portal. These are stored in an area accessible to the WOPS team.

For hosted CodeMeter License Central environments, a CmCloud FSB is used by default and integrated into the containers for the software vendor. A CmDongle FSB is available as an option for an additional fee.

**Wibu-Systems demonstrates best practices by using its own CodeMeter solutions internally. CodeMeter is comprehensive in its options and capabilities and may appear complex at first glance. This guide demonstrates that for all relevant use cases, there are clear and straightforward implementation paths that help transform complexity into flexibility.**

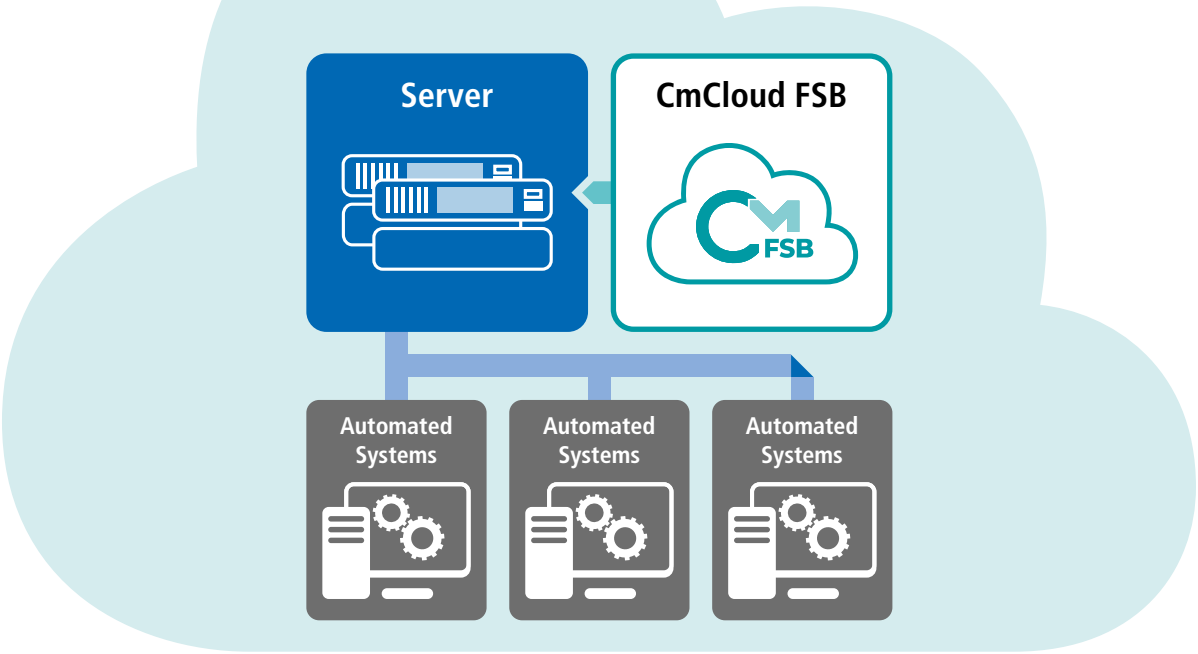



Figure 4: License Server for Cloud-Based Build Pipelines

# CodeMeter 9.00 – Architectural Transformation with Guaranteed Compatibility

With CodeMeter 9.00, Wibu-Systems initiates a fundamental transformation of the CodeMeter Runtime architecture. This process will be implemented step by step in upcoming releases and will culminate in a universal runtime environment that serves both traditional CmDongles and new (Post-Quantum Cryptography) PQC-enabled CmDongles (see the article https://www.wibu.com/magazine/keynote-articles/article/detail/pqc-for-software-licensing.html Post-Quantum Cryptography: Future-Proofing Your Software Licenses in KEYnote 50).



## Compatibility as the Top Priority

Wibu-Systems is fully committed to maintaining complete compatibility of the CodeMeter API and its behavior. Software that has been manually linked to the API or that uses CodeMeter Protection Suite requires no modifications.

However, the architectural changes do require adjustments on the software vendor side, particularly regarding runtime deployment. Three new services will gradually complement the environment, while certain functions of the existing CodeMeter license server (CodeMeter.exe / CodeMeterMacX / CodeMeterLin) will be separated. Command-line options for server control will also be affected.

In the future, installation and removal of services will be handled exclusively through installation programs or operating system tools, rather than via command-line switches. Network server configuration under Windows will continue to be managed through the installer. The cross-platform command-line option will be discontinued but can still be toggled at runtime using the cmu tool or CodeMeter WebAdmin. Full details are documented in the CodeMeter 9.00 Release Notes.

## CmLogger: The Future of Centralized Logging

CmLogger marks the first step in this transformation. As a new service, it initially takes over event logging for the license server.

Over time, it will evolve into the central logging instance, receiving messages from other services via secure channels.

Starting with version 9.00, CodeMeter installs with logging enabled by default, a long-standing request from many software vendors and consistent with industry best practices.

Log files rotate automatically, and older files are deleted to conserve storage space, while still remaining available for support cases. Timestamps are now recorded with millisecond precision in ISO 8601 format, including time zone information, enabling precise correlation across heterogeneous systems.

## Time Synchronization

A newly available option enables regular synchronization of certified time with CodeMeter time servers. Previously, this was only possible indirectly through a CmDongle backup option, which has since been discontinued.

Now, synchronization can be configured directly in CodeMeter WebAdmin (Settings) or via the cmu tool – ideal for automated post-installation configurations.

## Platform Changes

CodeMeter 9.00 no longer supports Windows 10 by default. For legacy operations, a paid long-term support option remains available with version 8.41.

The technically outdated Shell Extension (no longer installed by default since 8.20) has now been completely removed from the Windows installer.

## Universal Data for Everyone

The Product Item Option "Universal Data" is now also available for CmActLicense, and the CodeMeter license server supports it within CmCloudContainers.

Universal Data enables the use of asymmetric algorithms with longer key lengths and will also serve as a key storage location for Post-Quantum Cryptography in the future. In addition to storing key material with explicit algorithm specification, Universal Data can store data entries up to 64 KB per record.

Depending on configuration, software vendors may also write data at runtime.

Reading and writing data, as well as using keys, can be protected with passwords, which are themselves stored as Universal Data. Each Product Item provides an area with several thousand Universal Data entries, allowing multiple keys or datasets, including their associated passwords, to be stored without limitation.

## UserDefinedShare: Flexible License Sharing

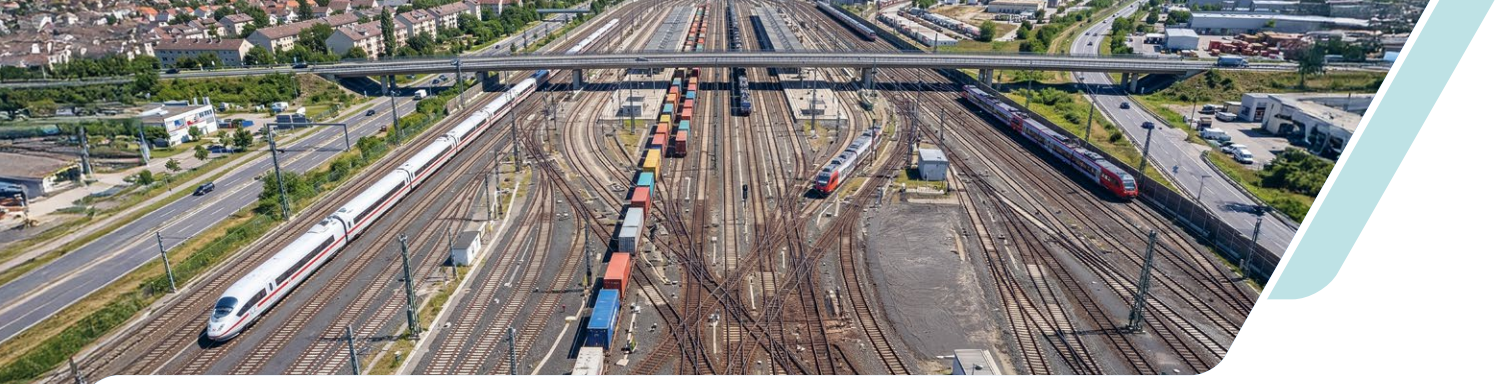
The new version introduces an additional option for license access modeled after StationShare license allocation. With StationShare, only one license is consumed per computer and identical license allocation parameters, even if the application is launched multiple times.

The new UserDefinedShare option also uses a single license when license allocation parameters are identical, provided the UserDefinedText parameter matches. If it differs, separate licenses are required.

This enables software vendors to define their own criteria for license sharing. The mechanism also works across multiple machines. Possible examples include the serial number or identifier of an external device, or the tenant ID in multitenant software to license concurrent usage of different tenants.

### Highlights of CodeMeter 9.00

- CmLogger modernizes logging: enabled by default, millisecond precision, automated rotation
- Universal Data expanded: now available for CmActLicenses & CmCloudContainers, up to 64 KB per entry with password protection
- UserDefinedShare: flexible license sharing based on custom criteria (serial number, tenant ID)



# Cost Optimization Through CodeMeter for Licensing

In economically challenging times, companies face pressure to identify new revenue opportunities while simultaneously uncovering potential cost savings. Developing new business models while conserving internal resources is a key objective of this challenge. One effective solution is the use of a professional licensing system such as CodeMeter.



## Protection and Licensing Are Not DIY Projects

*"We have our own software development department; we can easily implement software protection and monetization ourselves."* This statement is often heard in conversations at trade fairs or conferences. However, the full implications of such a claim are often misunderstood. What does it truly take to defend an application against hacker attacks and prevent illegal software use? What structures must be established and maintained within a company to seamlessly integrate a licensing system into existing process and achieve the highest level of automation? Do these structures need to be available globally, across different regions, and adapted to local requirements?

Such "homegrown" solutions require a significant investment of resources and personnel – resources that could be used far more profitably to strengthen the core competencies of the company's products. This is precisely where CodeMeter technology and Wibu-Systems' many years of experience in software protection and licensing come into play.

With our CodeMeter Protection Suite, protection and authenticity can be implemented through encryption for a wide range of target platforms and programming languages – not just for PCs, but also for various embedded platforms. This allows a manufacturer to protect and monetize different products using a single licensing technology. Valuable know-how, developed at considerable financial expense, is safeguarded, and by relying on a proven system such as CodeMeter, internal resources are preserved, both within development teams and within customer-facing support organizations.

## Compliance as an Opportunity

Today's compliance requirements for systems with digital components, such as the European Cyber Resilience Act (CRA), place software protection and licensing even more firmly in the spotlight and require action no later than by the end of 2027. On the one hand, security must be considered from the earliest stages of software design. On the other hand, product security must be documented and verifiable and, depending on the criticality of the industry, externally audited and certified.

Not everything needs to be developed in-house. CodeMeter technology supports key aspects such as software and data integrity, access control, data confidentiality, transparency of software versions deployed in the field, and ongoing enforcement of compliance throughout the product lifecycle. By leveraging a proven technology, companies can reduce internal costs while aligning compliance with a global monetization strategy.

## Where Licensing is Headed

Licensing has evolved significantly over time. While perpetual licenses were once the dominant model, the trend has increasingly shifted toward subscription-based approaches that generate recurring revenue over defined time periods. In addition, consumption-based licensing models are gaining traction. In these models, software or services are billed based on usage parameters, often calculated per unit of time, and sometimes with very fine granularity.

For software vendors, this means that licensing models must be designed to support such billing mechanisms from the start.


Cost savings can be achieved today by using a system like CodeMeter, which provides this flexibility out of the box and is continuously adapted to meet future licensing requirements.

Licensing is also increasingly moving to the cloud. While the ability to scale systems almost indefinitely offers major advantages, it also demands appropriate licensing models. This requirement applies not only to pure software vendors but also to the automation industry. Virtual controllers are already a reality today and bring these advantages into the industrial plants of the future. Licensing remains central to the monetization of virtualized control systems.

Anyone looking to deploy licensing models must remain operational across all deployment scenarios. Cost optimization can only be achieved if a company's diverse requirements are efficiently met using a single technology. CodeMeter supports offline scenarios through CmDongles or CmActLicenses, as well as cloud-based use cases using the same technology, without requiring any changes to the manufacturer's application.

## Cost Optimization Through Feature Enablement

Additional cost-saving potential also lies in production. For software-controlled hardware, higher-end hardware variants can be functionally limited through licenses. This reduces the number of product variants that need to be manufactured and lowers production costs. The licenses used to unlock existing features can later be transformed into additional revenue in the aftermarket.

Market examples include sensors with additional licensed features or drives where energy-optimized characteristic curves can be enabled via licensing. This trend is also evident in the automotive industry, where the term "Software-Defined Vehicle (SDV)" was coined many years ago. Entirely new business models are emerging as vehicle functions are activated through licenses purchased by the customer.

## Process Integration as a Central Factor

Significant cost-saving potential also lies in the efficiency of license management and its integration into existing internal processes. These processes must be highly automated, as any manual intervention in license management is extremely costly. This requires seamless integration of the licensing system into a company's existing back-office processes, so that, ideally, a customer order triggers license delivery with no manual interaction.

In this context, the licensing system must adapt to existing processes – not the other way around. Through connectors, gateways, and web services, CodeMeter License Central can be integrated into virtually any CRM or ERP system, as well as web shops.

Cost optimization potential also exists on the end-customer side. The easier the license activation is on the target system, the fewer support requests are generated. This includes service functions such as transferring a license to a new machine, reactivating a license after a hardware failure, and providing customers an overview of all their purchased licenses. These scenarios are covered by the CodeMeter License Portal as a self-service platform, enabling end customers to resolve most everyday cases independently.

In addition to direct end-customer interactions, license distribution through OEM partners can also be managed efficiently via CodeMeter License Portal. Manufacturers retain full visibility over all issued licenses and can allocate them to OEM partners for redistribution to end customers. A higher degree of automation is achieved when OEM partners are granted defined roles and permissions to generate specific licenses on demand. Integrated reporting then enables downstream settlement.

To further optimize licensing-related costs, Wibu-Systems offers these systems as hosted solutions. This shifts key operational responsibilities, such as availability, backups, updates, and security, to a professional service provider, resulting in additional cost savings. For globally operating companies, it is essential that hosted instances can be deployed worldwide using AWS services, while complying with regional requirements, such as those mandated in China.

### Increase Efficiency. Reduce Risk.

The use of CodeMeter across a company's entire licensing process leads to tangible cost reductions. Whether through flexible licensing models, seamless integration of license management into existing back-office systems, support for diverse market requirements using a single technology, or the tight integration of software protection and licensing to meet compliance demands, CodeMeter enables comprehensive cost optimization.

Anyone looking to implement software protection and licensing professionally needs a professional system. CodeMeter is the right solution.

# Software Subscriptions in Industrial Environments



## Business Model, Production Reality, and Technical Enforcement

Subscription models are well established across the software industry. In SaaS environments, they have effectively become standard. In industrial environments, however, the situation is more nuanced. While certain engineering software solutions have transitioned to subscription models, perpetual licenses with maintenance agreements continue to dominate machine-level or production-critical systems.

The primary reason is not the business model itself, but the technical and organizational framework of industrial IT. Production environments are often offline, safety-critical, and designed for maximum availability. In these settings, subscriptions must not introduce additional operational risk.

The key question, therefore, is not whether subscriptions make sense, but how they are deployed and maintained.

## Why Should Vendors Offer Subscriptions?

From a vendor's perspective, there are clear strategic arguments in favor of subscription models.

### Predictable, Recurring Revenue

Recurring revenue stabilizes cash flow, improves forecasting reliability, and positively impacts company valuation. Revenue is distributed more evenly over time, reducing dependence on large one-time projects.

### Higher Customer Lifetime Value

Instead of a single transactional license sale, subscriptions create an ongoing business relationship. Extensions, feature packages, and additional services can be systematically integrated.

### Lower Barriers to Entry for Customers

Spreading costs over time reduces upfront investment hurdles and accelerates purchasing decisions, especially for modular or feature-based products.

### Monetization of Services and Features

Predictive maintenance, analytics functions, remote support, or AI-driven enhancements are difficult to structure as one-time licenses. Subscriptions provide a flexible monetization framework for such offerings.

## Risks and Challenges for Vendors

Transitioning to subscriptions involves structural changes.

### Revenue Shift During the Transformation Phase

When moving from perpetual licenses to subscriptions, revenue timing shifts. This requires financial planning and clear communication with investors and stakeholders.

### More Complex System Integration

ERP systems, CRM platforms, billing processes, and license management must work consistently together. A subscription is not an isolated licensing model, but an integrated business process.

### Offline Reality in Industrial Environments

Production systems are often not permanently online. A subscription model must therefore avoid hard dependency on continuous cloud connectivity.

## Advantages and Disadvantages for Users

For industrial customers, operational reliability and predictability typically matter more than financial metrics.

### Reduced Upfront Investment

Instead of large one-time payments, costs are distributed over the subscription term. This lowers investment risk and facilitates budget approvals.

### Predictable Operating Expenses (OPEX)

Subscriptions create a consistent cost structure and can be fully expensed within the operating year.

### Continuous Development

Updates, security patches, and new features are typically in-

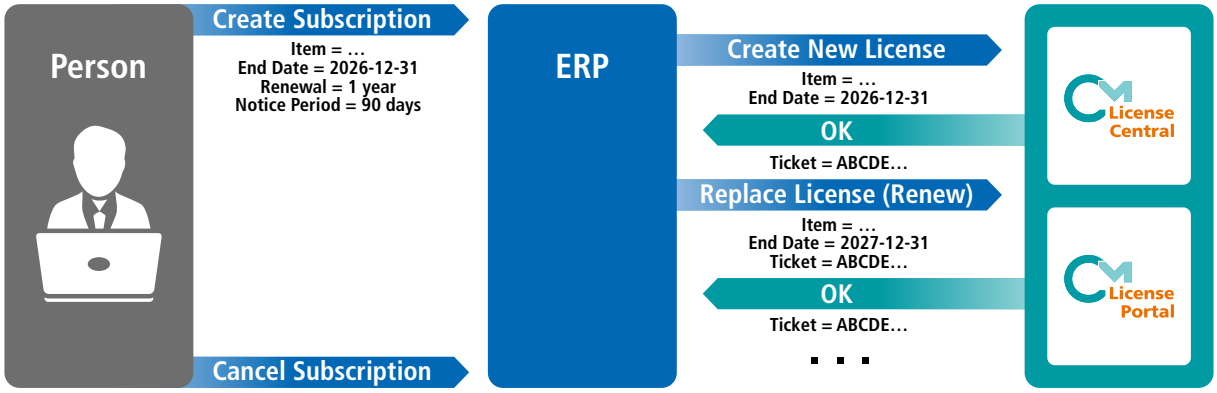



Figure 1: ERP Leading System

cluded. This reduces version fragmentation and simplifies support processes.

### Long-Term Total Cost

Over multiple years, a subscription may exceed the cost of a perpetual license. A transparent Total Cost of Ownership analysis is essential.

### Vendor Dependency

Subscriptions increase reliance on the provider – a strategic consideration in products with long life cycles.

### Transparency of Renewal Logic

Unclear renewal mechanisms or automatic renewals without transparent communication can quickly lead to acceptance issues.

### Technical Dependency on Licensing Infrastructure

Malfunctions during renewal or synchronization can have production-critical consequences.

Industrial systems are often:

- Air-gapped
- Only sporadically online
- Configured with strict security controls

Subscriptions will only be accepted if they function reliably and predictably even under offline conditions.

## Processes: Organizational Control of Subscriptions

In industrial contexts, a subscription is first and foremost a business process. The central question is: Which system manages the contract status? With CodeMeter License Central and CodeMeter License Portal, multiple architectures are possible.

### ERP System as the Leading System

The ERP system manages subscription terms, cancellation periods, and contract status.

When a subscription is created, the ERP system transfers the relevant term parameters to CodeMeter License Portal. The Portal generates a time-limited license with the corresponding expiration date.

At each renewal, the ERP system sends updated information to CodeMeter License Portal. The Portal adjusts the license term and provides the update to the user.

The ERP system remains commercially authoritative, while the Portal handles technical implementation.

### Asynchronous Control

The ERP system defines the start date, term, and renewal interval.

The initial process is identical to the "ERP system as the leading system" model: when a subscription is created, the ERP system transmits the relevant term parameters to CodeMeter License Portal. CodeMeter License Portal then generates a time-limited license with the same expiration date.

In contrast to the "ERP system as the leading system" model, however, CodeMeter License Portal renews the subscription automatically and asynchronously and provides the update to the user. Automatic renewal continues until the ERP system transmits a cancellation of the subscription.

Additionally, CodeMeter License Portal can notify the ERP system of the automatic renewal if this information is required for billing within the ERP system.

### Management Within CodeMeter License Portal

Alternatively, CodeMeter License Portal itself can act as the leading system. Start and end dates are managed there. The ERP system periodically receives billing data and is responsible only for invoicing. Online payment providers can also be integrated directly with CodeMeter License Portal, with the ERP system optionally receiving the data for informational purposes.

This approach decouples contract logic from ERP system complexity. It is particularly relevant in industrial environments where ERP systems are designed primarily for one-time hard-

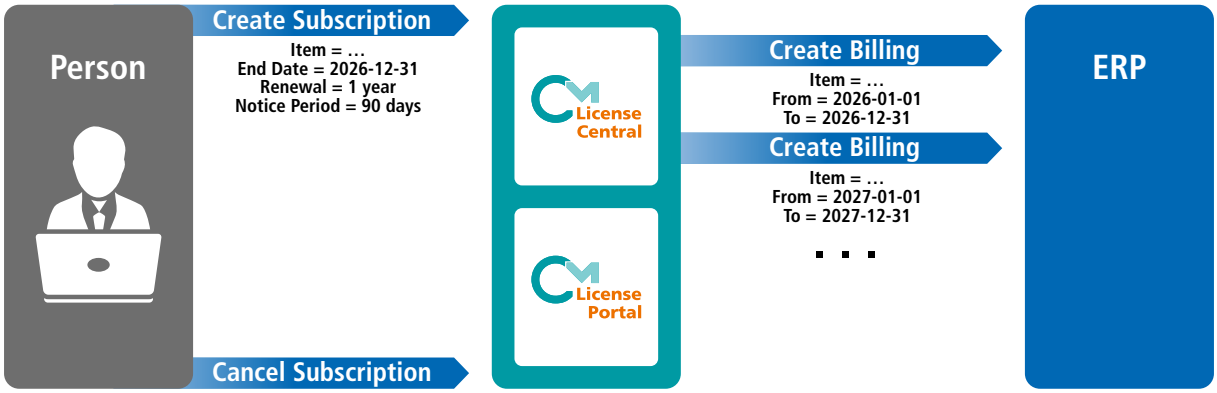



Figure 2: License Portal Leading System

ware sales. Modifying ERP systems often involves long implementation timelines and significant cost, making subscription management within CodeMeter License Portal an attractive alternative.

### Time Quotas

Strictly speaking, time quotas are not subscriptions. They are prepaid models in which the user purchases a defined time package. Once expired, a new quota is purchased and activated.

This model is frequently used to simplify initial onboarding, with the option to transition into a full subscription model while crediting any remaining time.

## Technical Enforcement: Enforcing the Subscription Term

While the previous section addressed organizational aspects, this section focuses on technical implementation.

Technically, a subscription is implemented as a **time-limited license**. Licenses within the license container are assigned individual expiration dates. Activation follows the same process as a traditional license. The user receives a ticket and activates it on the target system. In online scenarios, this occurs with a single click via a Software Activation Wizard that uses the License Portal API.

In offline scenarios, a context file is generated from the target container. This file contains a fingerprint and the licenses already present. Using this context file and the ticket, activation is performed on an online system within CodeMeter License Portal. The resulting update file is transferred to the offline system and imported into the target container. Optionally, a receipt file can be generated and uploaded to CodeMeter License Portal.

Upon renewal, the expiration date is updated. CodeMeter License Portal recognizes the target container and generates the appropriate update file accordingly. In online scenarios, the Software Activation Wizard automatically downloads and applies the update in the background. In offline scenarios, the user downloads the file from CodeMeter License Portal and transfers it manually to the offline computer.

An optional approach is to preload a base license during production onto a newly manufactured device. This creates the license container and eliminates the need to generate a context file on the target system during initial activation.

Even during operation, transmitting a new context file can be beneficial. A new context file confirms that updates have been received and reduces the size of subsequent update files. If a subscription needs to be transferred to another device, the software vendor determines whether a receipt-based confirmation from the original device is required, or whether limited self-service transfer without receipt is permitted.

### Internal Subscriptions

Subscriptions are often associated with monetary transactions, but this is not necessarily the case. There are internal scenarios in which subscription-like mechanisms are useful.

Licenses used by employees and sales partners for demonstrations are often structured as subscriptions. If a license is lost, renewal can simply be discontinued. In some cases, renewal periods are intentionally shorter than for customer subscriptions.

Subscriptions are also valuable for test licenses within development or QA departments. Terms can be configured in minutes or hours to ensure sufficient licensing during testing, while also enabling validation of expired-license scenarios.

**Subscriptions can function effectively in industrial environments when implemented, as with CodeMeter, to reliably support offline scenarios above all else.**

# ENFORCERS: Enhanced Cooperation for Cybersecurity

Industrial automation and manufacturing are undergoing a profound transformation. Increasing connectivity, long product lifecycles, heterogeneous Operational Technology (OT) environments, and strict availability requirements make cybersecurity in industry fundamentally different from IT security. Detecting incidents is no longer enough: vulnerabilities must be mitigated, software updated, and trust restored across complex supply chains, often under real-world constraints such as segmented networks or limited connectivity.

These realities formed the backdrop for ENFORCERS, a European project launched to address cybersecurity as a lifecycle challenge. At the core of ENFORCERS is the development of a Cybersecurity System Platform that connects these processes into a cooperative framework. Private Security Operation Centers (SOCs) play a central role in collecting and correlating vulnerability data, while Digital Elements anchored in Secure Elements establish trust at critical OT edges and gateways. Automated workflows link incident handling with secure updates and re-certification, ensuring that software integrity can be restored even in complex industrial environments. The approach directly supports compliance with NIS2 and anticipates the requirements of the Cyber Resilience Act, while remaining adaptable to future regulatory and cryptographic developments.

ENFORCERS brings together a strong consortium of industrial, technology, and research partners from across Europe. Industrial companies such as Balluff (Germany and Hungary), Schneider Electric (France), TTTECH Computertechnik (Austria), and Nexus Secured Business Solutions (Sweden) contribute real-world requirements from automation and manufacturing. Cybersecurity and technology specialists including Infineon Technologies (Germany), Langlauf Security Automation (Germany), DYNAMIKI (Greece), AITAD (Germany), and ResilTech (Italy) provide expertise spanning secure elements, cryptography, SOC integration, and incident response. Applied research is supported by Fraunhofer SIT, while VDMA contributes its industrial network and perspective on standardization and adoption.

The project is co-funded by the European Union under the Digital Europe Program and supported by the European Cybersecurity Competence Centre, underlining the EU's role in enabling cross-border cooperation on cybersecurity challenges. ENFORCERS is planned as a three-year initiative, with clearly defined phases covering requirements analysis, system architecture, implementation, demonstration, and dissemination.

The project's official kickoff took place on 10–11 February 2026 at Wibu-Systems' headquarters in Karlsruhe, where consortium partners met in person to align on objectives, responsibilities, and the technical roadmap. As project coordinator, Wibu-Systems led the discussions, marking the transition from concept to execution and setting the foundation for close collaboration in the years ahead.

READ MORE

ECCC
EUROPEAN CYBERSECURITY COMPETENCE CENTRE

Co-funded by the European Union

# Copy & Start – No CodeMeter Runtime Needed, now also for Windows Intel & ARM

In certain scenarios, it is desirable for an application to work directly with a local CmDongle without requiring prior installation of CodeMeter components. This is now possible, without any modification to the software, using AxProtector NC for native Windows applications on both Intel and ARM architectures.



## CPSRT as a Universal Execution Environment

AxProtector safeguards applications using state-of-the-art protection technologies and integrates them seamlessly into the CodeMeter licensing system. The license keys required for decryption are securely stored, whether in a hardware-based CmDongle, a software-based CmActLicense, or a cloud-based CmCloudContainer. Connectivity is managed flexibly via the WibuCm library in combination with CodeMeter Runtime.

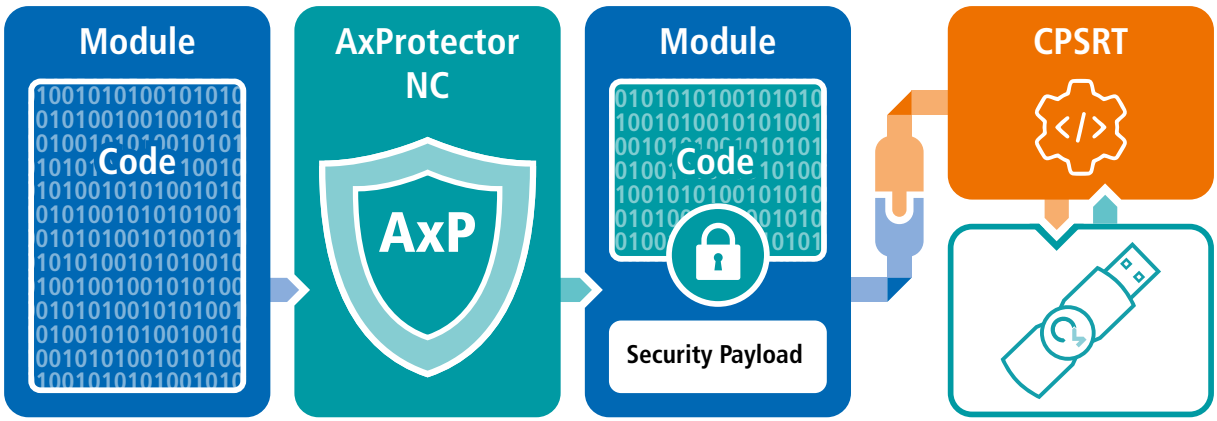
Previously, AxProtector used different execution environments depending on the application type, such as native C++ programs or .NET assemblies. With the expansion to Python and JavaScript applications, the goal was to create a unified, future-proof approach.

The result is CodeMeter Protection Suite Runtime (CPSRT), a central native protection component that combines program encryption, license verification, and anti-reversing measures within a single module. By default, integration with CodeMeter licensing is achieved through an installed CodeMeter Runtime. Alternatively, locally connected CmDongles can be used directly, entirely without installing additional software; only the accompanying CPSRT is required. This optional, flexible mode allows for easy deployment of applications on customer devices or mobile workstations, with no setup effort or administrative privileges needed.

CPSRT forms the technological foundation for AxProtector NC (Native Core) and supports .NET, Python, and JavaScript applications, Linux binaries, and AxProtector CTP (Compile Time Protection). Developers benefit from a unified, high-performance protection solution that delivers maximum security while significantly simplifying software deployment and use.

## Windows on ARM

Microsoft Windows is the world's leading desktop operating



system and was long closely associated with Intel-compatible processors. Following its success in embedded and mobile environments, the ARM architecture has now established itself in the desktop space as well. Windows is therefore also available natively for ARM.

An integrated emulation layer enables dynamic translation of Intel-compiled applications into ARM code, allowing many programs to run without recompilation. Intel applications previously protected with AxProtector can also be executed in compatibility mode in this way, provided they were protected with limited anti-reversing measures.

Despite this strong compatibility, demand continues to grow for AxProtector protection specifically for native Windows binaries. AxProtector NC already delivers solutions for .NET, Python, and JavaScript programs, as well as for native applications via CTP. A CPSRT version compiled specifically for Windows on ARM enables protection without an emulation layer, improving both performance and stability.

## AxProtector NC for Windows Intel and ARM

With the native support of Windows on ARM comes the expectation that protection mechanisms should also be available across architectures. AxProtector NC addresses this need: Starting with release 11.80 of the CodeMeter Protection Suite, CPSRT-based technology is available for compiled Windows binaries on both Intel and ARM architectures.

Applications can thus be protected consistently, regardless of processor architecture. They run natively, efficiently, and without reliance on emulation mechanisms. AxProtector NC for Windows supports the established features of AxProtector Native Core for .NET, Python, JavaScript, Linux binaries, and CTP. These include automatic encryption, flexible licensing, file encryption, and the outsourcing of sensitive code sequences (CodeMoving). Applications can also run directly with locally connected CmDongles, without requiring installation of CodeMeter Runtime.

The new automatic function encryption additionally enables modular licensing of individual program components without requiring changes to the source code.

With its expansion to native Windows binaries, AxProtector NC provides a unified solution for secure and efficient application protection across all Windows architectures.

- AxProtector NC supports local CmDongle licensing without an installed CodeMeter Runtime, thanks to CPSRT with integrated CodeMeter Embedded.
- CPSRT serves as a unified execution environment for .NET, Python, JavaScript, and native applications.
- AxProtector NC already supported .NET, Python, JavaScript, Linux binaries, and CTP.
- AxProtector NC version 11.80 supports Windows binaries for both Intel and ARM architectures.
- All NC features are available, including Automatic Protection, Modular Licensing, Unbound Mode, CodeMoving, and File Encryption.

# CodeMeter Embedded: Security Down to the Microcontroller

Is the firmware of sensors, controllers, or drives protected against copying and attacks? CodeMeter Embedded protects intellectual property (IP) and implements licensing seamlessly, all the way down to the microcontroller level. Manufacturers can deploy CodeMeter technology specifically for embedded environments.

The valuable know-how embedded in the firmware of small devices such as controllers, frequency inverters, or sensors is vulnerable to attacks on integrity. This affects both the firmware itself and the licensing models in use. Wibu-Systems extends CodeMeter Embedded to microcontrollers and real-time operating systems (RTOS) to deliver future-proof, crypto-agile solutions. Manufacturers can protect their IP, implement licensing, and benefit from end-to-end compatibility across the CodeMeter ecosystem.

## Challenges in Embedded Systems

Firmware in industrial sensors, controllers, or drives contains trade secrets that are exposed to the risk of reverse engineering. Cyberattacks can compromise individual components or, if these are used as entry points, endanger entire machines and industrial plants. While the potential damage in isolated machines may still be manageable, it becomes significantly more severe in critical infrastructure (KRITIS) environments.

Microcontrollers, particularly those running a single application or an RTOS, present challenges for secure protection and licensing due to limited resources such as memory, processing power, and communication interfaces. Integrating a cryptographic library for signatures or encryption may seem simple at first, but it quickly becomes complex when considering the overall environment:

- Where are keys stored securely?
- How are keys and key updates delivered to the device?
- How are licenses provisioned to the device?
- How are keys and licenses managed centrally?
- How is data transferred when devices are not networked at all or only via specialized interfaces?

In production facilities or KRITIS environments, devices are often isolated from the public Internet. As a result, online-hosted license management systems and traditional cloud solutions are not viable options. These constraints affect not only microcontroller-based systems, but virtually all embedded platforms used in industrial automation, medical technology, and comparable control and regulation applications.

## CodeMeter Embedded on MCUs and RTOS

CodeMeter Embedded is a compact library providing CodeMeter API functions for embedded operating systems such as Linux, VxWorks, or QNX. For many microcontroller platforms, however, even a few hundred kilobytes can be too large, and features such as caching, shared memory, or network access are often unnecessary. This is where CodeMeter's modularity comes into play: the code can be configured to run even on severely resource-constrained microcontrollers in bare-metal environments or in combination with an RTOS.

The well-known CodeMeter Embedded package is delivered as a source code package, allowing developers to compile it for a wide range of microcontroller platforms. The CodeMeter API remains fully compatible with the entire CodeMeter API ecosystem, enabling familiar function calls to be seamlessly ported to the microcontroller level. Unlike OS-based CodeMeter Embedded deployments, developers must implement application-specific adaptations for MCUs, such as storing licenses in defined flash memory areas, enabling remote updates, or connecting to a CodeMeter ASIC via SPI.

Most importantly for manufacturers, the CodeMeter API and license formats remain the same as those used in PC environments. This creates a licensing system at the microcontroller level that is fully compatible with larger embedded systems, as well as PC and server platforms. CodeMeter License Central can issue licenses consistently for both powerful PC systems and small microcontrollers, keeping production, rollout, and update processes consistent. Software manufacturers and firmware developers select the appropriate CodeMeter Embedded variant from a modular toolkit and integrate it directly into the application to be protected.

## Crypto Agility and Post-Quantum Cryptography (PQC)

Currently, asymmetric cryptographic algorithms have not yet been broken by quantum computers. However, such a breakthrough could happen at any time and without warning. Even if initially limited primarily to academic relevance, each decryption attempt would require deployment of quantum resources – resources that will likely be applied where economic incentives are high, such as for copying highly valuable software.

Once the value of the protected data exceeds the cost of a quantum attack on the keys in use, protection strategies must

be updated. For this reason, PQC is currently being incorporated into all CodeMeter products, ensuring that fully PQC-capable software is already deployed in the field on day one. The next generation of CodeMeter dongles will support longer PQC keys and modern PQC algorithms such as ML-KEM and ML-DSA. Crypto agility has been built in to allow seamless algorithm transitions using existing hardware and software.

For CodeMeter customers, this means they do not need to develop complex PQC strategies themselves. Instead, they benefit from a platform that has already made the necessary preparations and continues to evolve. Where PQC cannot be deployed directly due to key length or computational constraints, such as on very small microcontrollers, hybrid approaches are used to ensure secure protection and licensing.

## Regulatory Requirements and Security Standards

Organizations are subject to strict standards for secure software in systems and components, with each economic region defining its own regulations and compliance frameworks. CodeMeter Embedded and complementary products such as CodeMeter Certificate Vault help software vendors and system integrators meet these requirements efficiently through tamper-proofing, crypto agility, license compliance, and the secure use of certificates.

Key standards include:

- **Worldwide**
  - IEC/ISA 62443
- **Europe**
  - Cyber Resilience Act (CRA)
  - NIS2 Directive
- **USA**
  - NIST Cybersecurity Framework (CSF 2.0)
  - CISA Cybersecurity Performance Goals (CPGs)
- **China**
  - Grading Protection 2.0 (MLPS 2.0)
  - CCoP 2.0 (Cybersecurity Compliance for CII)
  - GB Safety Standards (2026)

DEMO

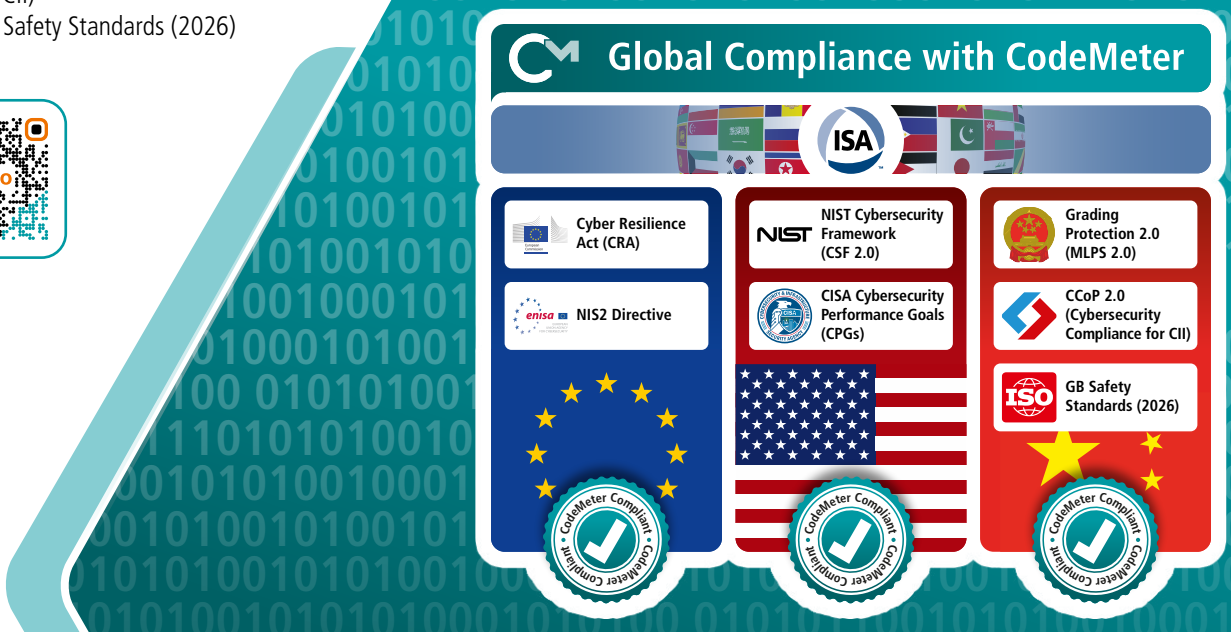
CodeMeter addresses core requirements such as risk management, integrity, and secure updates, facilitating certification and enabling standardized security-by-design approaches in embedded systems.

## CodeMeter Portfolio at a Glance

CodeMeter Runtime is a service that runs on desktop PCs and high-performance embedded systems, while CodeMeter Embedded is integrated as a software library directly into the application being protected. Both use the same CodeMeter API, the same license format, and store licenses and keys in CmContainers (CmDongles, software-based CmActLicenses, CmCloudContainers). They are fully compatible with CodeMeter License Central and support a wide range of operating systems and hardware platforms, down to the microcontroller level.

CodeMeter Protection Suite complements this approach by embedding high-security IP protection directly within the application. This enables manufacturers to combine licensing, copy protection, and know-how protection within a unified technical and organizational framework.

CodeMeter Embedded lays the foundation for a unified security and licensing concept across all device classes. From desktops to bare-metal MCUs, the same mechanisms for licenses, keys, and cryptography apply, all supported by a crypto-agile architecture. This makes heterogeneous embedded landscapes manageable and allows manufacturers to focus their development resources on core competencies and new services built around their connected devices.



# Stateless Apps, Stateful Licenses: A Strategic Guide to Containerized Protection

The rise of containerization has fundamentally altered the software delivery lifecycle. For Independent Software Vendors (ISVs), the transition to Docker and Kubernetes offers significant scalability advantages. However, it also introduces what can be described as the Containerization Paradox: containers are ephemeral and stateless, while traditional licensing is rooted in persistent hardware binding.



To bridge this gap, technical decision-makers are advised to move beyond "local" activations and architect a licensing strategy that respects the nature of the container.

## The Implementation Landscape: A Scenario-Based Map

A one-size-fits-all approach is rarely effective in containerized deployments. The following map helps align your infrastructure with the correct CodeMeter architecture.

## Quick Wins: SmartBind – Local Binding in Docker

In the early stages of container adoption, many ISVs attempted to run the CodeMeter Runtime directly inside the container using **SmartBind**. While this is an attractive "low-effort" path for a Proof of Concept (PoC), it may prove technically fragile in production environments.

The hardware fingerprints that SmartBind relies on (MAC addresses, CPU IDs, disk UUIDs, etc.) are often abstracted or virtualized by the Docker engine. In a containerized context, these identifiers can become less reliable. If a container is rescheduled to a different node in a cluster, the binding breaks. Furthermore, binding inside the container offers limited security, as the quality of the license "anchor" is generally lower than on a bare-metal host.

**The Technical Verdict:** SmartBind can be appropriate for PoCs, while host-bound licensing is recommended for production.

## CmCloud for SaaS: Identity-Centric Licensing

As ISVs shift toward subscription-based SaaS models, the traditional machine-binding model can become limiting. **CmCloud** effectively redefines the "container" by moving the license identity to a secure Wibu-Systems cloud server.

| Deployment Scenario | Recommended Architecture | Strategic Rationale |
|---|---|---|
| **Public Cloud / SaaS** | **CmCloud / CmCloud Lite** | Complete hardware decoupling; identity-based Licensing. |
| **Private Cloud (On-Prem)** | **CmDongle or Network Server** | Centralized entitlement management within a customer's secure LAN. |
| **Air-Gapped Environments** | **CmDongle or Local Network Server** | High-integrity protection without external "heartbeat" requirements. |
| **Edge & Industrial OT** | **Host-Bound / CmASIC / CmDongle** | Physical host as a root of trust for containerized logic. |

Implementation Landscapes

```
services:
  app:
    image: app-image:latest
    environment:
      CODEMETER_HOST: codemeter
  codemeter:
    image: "wibusystems/codemeter:latest"
    environment:
      CM_REMOTE_SERVER: host.docker.internal
    extra_hosts:
      - "host.docker.internal:host-gateway"
```

Figure 1: Connecting to a Host-Bound Network Server

```
services:
  app:
    image: app-image:latest
    environment:
      CODEMETER_HOST: codemeter
  codemeter:
    image: "wibusystems/codemeter:latest"
    environment:
      - CM_CMCLOUD_CREDENTIALS=/etc/wibu/cloud/cred.wbc
    volumes:
      - ./secrets/cred.wbc:/etc/wibu/cloud/cred.wbc
```

Figure 2: Licensing Setup with CmCloud

**The Concept:** The application container no longer holds a license; it carries a **Credential**. This credential acts as a secure key to a specific CmCloudContainer hosted in the cloud.

- **Complete Decoupling:** The license follows the instance or user rather than being bound to specific hardware. Containers can be killed and recreated across hundreds of different hosts without re-activation.
- **High Awareness:** ISVs gain real-time visibility into license usage, enabling data-driven decisions on feature adoption and seat allocation.
- **SaaS-Ready:** Aligns naturally with modern web-based deployments where "hardware" is a fluid concept.
- **The Trade-off:** Requires a persistent Internet connection to maintain the cryptographic "heartbeat" with the cloud server.

## The Reference Architecture: Decoupled Network Licensing

For on-premise Docker or Kubernetes clusters, it is advisable for the license to reside **outside** the application container. This creates a "Stateless App / Stateful License" architecture.

- **Resilience:** If a container is rescheduled by an orchestrator, it simply re-establishes a connection to the Network Server.
- **Separation of Concerns:** The application image remains focused on business logic; the CodeMeter License Server (on the host or a separate VM) manages the cryptographic state.

## Edge Computing: Establishing a Root of Trust

In Industrial IoT (IIoT), you must protect IP on decentralized hardware. The **Hybrid Host Model** represents a widely adopted technical approach: the CodeMeter Runtime is installed on the **Host OS**, and containers access it via an internal network bridge.

### Path A: Hardware-Bound Virtual Licenses

Best for mass-scale deployment on standardized hardware. The license is bound to the physical machine's fingerprints but managed as a secure file on the host. This combines "soft" delivery with "hard" host-level security.

### Path B: CmDongle and CmASIC (the Physical Enclave)

For high-assurance security, a physical hardware enclave is used. **CmASIC** is integrated into the printed circuit board (PCB), while CmDongle provides USB or SD card-based hardware. The host Runtime manages the communication with the chip, while the containerized app consumes the license via the internal bridge, ensuring cryptographic keys never leave the silicon.

## Strategic Guardrails: Recommended ISV Practices

- **DON'T bind virtual licenses inside the container.** Always set up virtual licenses on the **Host OS** to ensure a high-quality, stable binding.
- **DON'T mount USB devices directly to the application container.** Set up a license server on the host and access licenses via network connection.
- **DO leverage CmCloud** for friction-free scaling in connected environments.
- **DO leverage network licenses** for maximum flexibility in air-gapped environments.
- **AVOID high-frequency activation cycles.** Do not reactivate SmartBind licenses for every ephemeral container spin-up.

### Licensing as an Enabler

In a containerized world, licensing should not be viewed merely as a hurdle, but as a strategic layer of the application architecture. By moving away from potentially fragile internal container bindings and toward decoupled network or cloud models, ISVs can provide the flexibility their customers demand while maintaining the security their business requires. The objective is to achieve a "Protection-by-Design" approach where the license is as scalable and portable as the code it protects.


READ MORE

# Licensing-as-a-Service for Rapidly Innovating Software Companies

Monetization is now the primary means of driving revenue growth in digitalization strategies. Time-to-market is a critical competitive advantage when introducing software. At the same time, licensing should not become an obstacle, but rather an enabler. Licensing-as-a-service is the solution.



The strength of your company lies in its innovative power to develop market-specific solutions. This is the key to your success and allows you to ride the wave of growth. But how can this financial success be realized? On the one hand, your applications must be protected against reverse engineering. On the other hand, you want to introduce common licensing models that are flexible and aligned with markets and customers.

The challenges you face are diverse. You need to bring products to market quickly, scale globally, and support a variety of business models, from perpetual licenses and subscriptions to trial versions and feature-based offerings – preferably also for after-sales. At the same time, licensing must function reliably on the customer side in offline systems, network environments, and cloud-based deployments. A protection and licensing system like CodeMeter has successfully delivered exactly this for many years, tailored to the diverse requirements of customers. Building such a flexible and comprehensive system offers maximum control, but requires time and internal resources to design and integrate it into your company's back-office structures. For entering the world of licensing, however, what is needed is a simple, easy-to-use system for both software vendors and their customers.

This is exactly what CodeMeter Licensing-as-a-Service (CmLaaS) is designed to address. The goal in developing this product was to make the entire CodeMeter ecosystem immediately and easily accessible. No deep dive into the underlying technology, no time-consuming definition of license models at the lowest level, and no complex processes for generating and distributing licenses. Everything is predefined and ready for immediate use. A single access to CmLaaS is sufficient to implement software protection and monetization quickly. Predefined license models, such as trial licenses, internal licenses, perpetual licenses (local or network-based), and subscription models, are already available as templates and can be used right away. CmDongles, CmActLicenses, and CmCloudContainers are directly supported as target license containers.

Within the underlying CodeMeter License Central, which serves as the foundation of the service, everything is already in place. For complete license management, from creation and control to activation on the customer side, CodeMeter License Portal is also available as part of CmLaaS, with all its advantages. This enables your customers to manage their licenses independently via the self-service portal.

CmLaaS grows with your success. If you want to distribute licenses globally, not just on your own but also through resellers, this can be done via the Portal. You can set up a reseller, allocate licenses for redistribution, or even grant them permission to generate specific licenses independently. These can then be tracked through reporting. In all cases, you maintain full control over the licenses issued.

The major strength of CodeMeter technology lies in the seamless interaction between software protection and licensing. CodeMeter Licensing-as-a-Service supports this by generating a control file for AxProtector, based on the selected license template, which is used to create a protected application for a specific target system or programming language. This allows protection and licensing to be combined quickly and efficiently with minimal effort.

From a cost perspective, CmLaaS is fully aligned with the ramp-up strategy of highly innovative software companies.

It offers very low entry costs, minimal ongoing expenses, and minimal resource requirements, all while still leveraging the full benefits of CodeMeter technology.

For those seeking a cost-efficient solution with rapid time-to-market, and wanting to combine protection and licensing without compromise, will find the answer in CodeMeter Licensing-as-a-Service.

### CodeMeter Licensing-as-a-Service

- Simplest entry into software monetization
- Significantly reduced time-to-market for your products
- The ideal combination of software protection and licensing
- Minimal entry costs with full functionality
- Upgrade path to a proprietary licensing system




## One idea at the right time can change everything.

**Subscribe to our blog**

# Should Software Licensing Always Follow the Latest Trends?

Certificates and JSON Web Tokens are just two of the many technologies I have encountered over the past 23 years at Wibu-Systems working in software protection and licensing. Interestingly, these suggestions did not come from my colleagues but from customers, who asked why we were not using these technologies and suggested that things could be much simpler if we did.



CodeMeter has been on the market for more than 20 years. Our customers – Independent Software Vendors (ISVs) – use CodeMeter to protect applications that often have similarly long lifecycles. For them, continuity, compatibility, and steady improvement are far more important than chasing the latest trends.

## Our Motivation

Becoming best-in-class comes from experience and from putting that experience into practice. When we created CodeMeter, and especially the Universal Firm Code, our architecture team already had several hundred years of combined hands-on experience in applied cryptography, particularly in software protection and licensing.

It is therefore no surprise that many of the fundamental principles found in the technologies mentioned earlier were already implemented in CodeMeter long before those technologies even had the names we know today. After all, they all rely on the same cryptographic foundations: symmetric encryption, asymmetric encryption, hash functions, and more.

Of course, it is gratifying to see that CodeMeter has followed a structure similar to JWTs for many years and internally uses certificate chains. But that is no reason to become complacent. Remaining best in class requires constant questioning, continuous practice, and ongoing improvement.

## Behind the Scenes

"Encryption is the key." This could easily describe the guiding principle behind CodeMeter. To effectively protect software against reverse engineering, modification, and piracy, CodeMeter relies on encrypting the application itself. This can include executable files as well as data files.

In textbooks, one often finds the following simplified recommendation, for example when sending encrypted emails. If data is to be securely transmitted from a sender (S) to a recipient (R), R first generates an asymmetric key pair and sends the public key to S. S encrypts the data using a ra ndomly generated session key and then encrypts that session key with R's public key. The encrypted data and the encrypted session key are then sent to R. Only R can decrypt the session key, and therefore the data itself.

### Practice vs. Textbook Theory

In software protection, however, two aspects contradict from the standard textbook model.

- First, while the recipient of an email has a strong interest in keeping their private key secret, the user of a software application often has no such incentive.
- Second, if the same email content is later sent to another recipient, a new session key is used. With software, which is often large and distributed through multiple channels, this approach is impractical. It would require providing each user with a separate program encrypted with a different session key.

### A Solution Inspired by the Textbook

For this reason, CodeMeter generates a key, similar to a session key, to encrypt the software. In simplified terms, this key

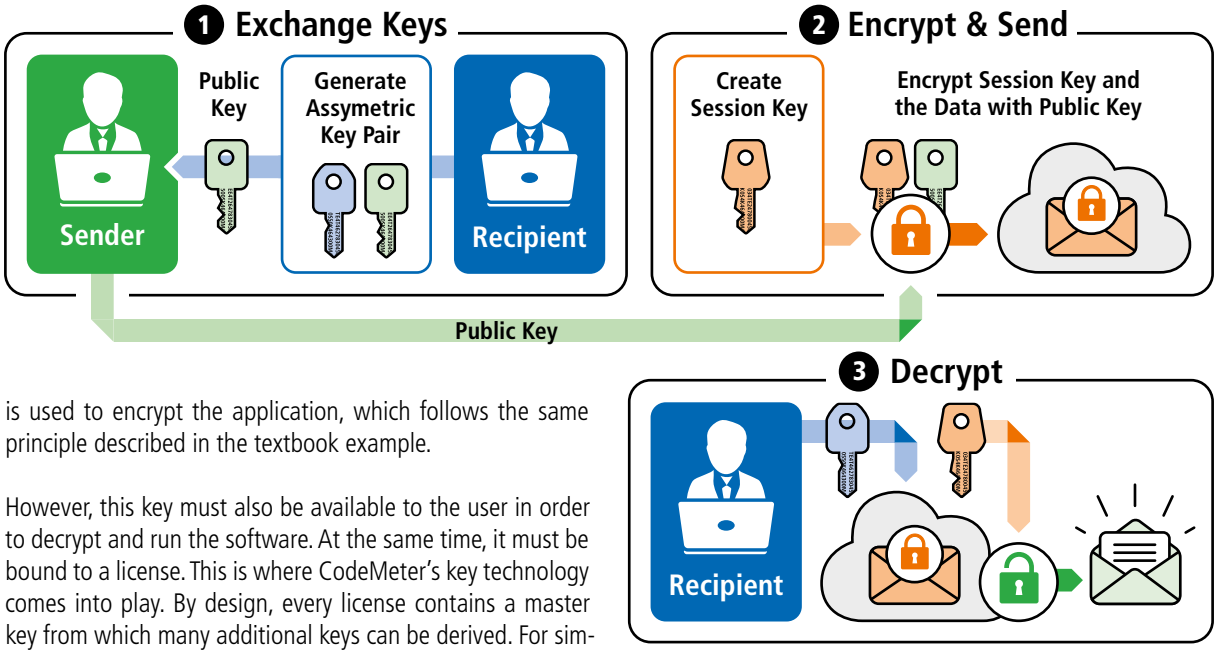


Figure 1: Sending Encrypted Emails with Asymmetric Keys

is used to encrypt the application, which follows the same principle described in the textbook example.

However, this key must also be available to the user in order to decrypt and run the software. At the same time, it must be bound to a license. This is where CodeMeter's key technology comes into play. By design, every license contains a master key from which many additional keys can be derived. For simplicity, let's assume there is one key per license. This key is delivered to the user together with the license – again following the textbook principle.

CodeMeter generates a key pair in the target container. For a CmDongle, this happens during manufacturing. The public key is then sent to the software vendor in the form of a context file. CodeMeter uses this public key to encrypt the license – most importantly, the key contained within it. The license, including the key, is then delivered to the user, where it is decrypted and securely stored within CodeMeter.

With a CmDongle or a CmCloudContainer, this happens inside the dongle or in the cloud, hidden from the user. With a CmActLicense, the process takes place inside CodeMeter Runtime, and hardware properties of the target device are used to generate the key pair. More details about this mechanism can be found under SmartBind. In the actual implementation, an additional internal mechanism is used for security and performance reasons, which is beyond the scope of this article.

### Is the License Genuine?

Securely transferring the keys required to run the software provides the highest level of protection for an application, except in the case of CodeMoving. However, another important question remains: Is the license itself genuine?

An attacker could generate an update file and encrypt it with the public key of the target container. The container must be able to detect such a fake and reject the contents of the update.

To prevent this, updates are digitally signed with a private key. The corresponding public key is stored in the target container and used to verify the signature. Each software vendor has a unique signing key. To ensure that the public key in the container cannot be forged, CodeMeter relies on mechanisms similar to certificate chains.

### Is the Target Container Genuine?

A genuine target container will reject a fake update. However, CodeMeter must also avoid issuing updates to a fake container. If this were possible, a fake container could decrypt the data, including the keys it contains.

To address this, CodeMeter again relies on certificate-like mechanisms. With a CmDongle, a certificate for the public key is created during manufacturing and stored inside the dongle. For a CmCloudContainer, this process is securely handled in the cloud using a key pair unique to each cloud system.

For a CmActContainer, the process occurs within CodeMeter Runtime on the user's target system. From a cryptographic perspective, this is the weakest link in the chain. At this point, the principle shifts to "trick, deceive, and camouflage," since the standard textbook approach can no longer be applied directly.

## JSON Web Token (JWT)

Returning to the original question of why we do not simply use certificates and JWTs: a JSON Web Token consists of a header, a payload, and a signature. CodeMeter licenses follow a similar structural principle, but CodeMeter uses a different format.

In most discussions about JWT, people are actually referring to JWS (JSON Web Signature). However, JWS is not suitable for transmitting confidential information such as private keys.

For transmitting confidential data, JWE (JSON Web Encryption) is used instead. Best practice is typically to nest a signed JWT inside a JWE so that the data is both signed and encrypted.

In principle, this is exactly the approach used by CodeMeter: the license data (the update file) is both signed and encrypted.

Another key difference is that CodeMeter provides the entire key management infrastructure required for licenses. When using JWE, developers must implement this part themselves. Tokens usually have an expiration date, which requires regular renewal. In offline scenarios, this can complicate things for the user. If the expiration period is set far into the future, revocation becomes difficult. While CodeMeter simulates its own internal clock, tokens in offline scenarios are verified against the system time of the target device. How resistant that is to manipulation is something I leave for the reader to judge.

For very small systems, or systems where encryption is not required, we have already combined CodeMeter with JWT. In these cases, the keys required to generate JWTs are stored within CodeMeter. A usage counter can define the authorization so that only a limited number of JWTs can be generated. This allows the JWT to be created on the user side, even in an untrusted environment. In our projects, the payload typically consists of the device fingerprint and the license information.

## Certificates

The principles behind certificates are also used in many areas within CodeMeter. However, certificates themselves are not intended for transporting confidential data, and due to the ASN.1 format, they are often relatively bulky.

In a purely certificate-based solution, a key pair would typically be generated on the target device. The public key would then be sent to the software vendor, who would issue a certificate that adds the device to the set of trusted systems. Licensing, or the equivalent entitlement logic, would still need to be implemented separately.

Binding the certificate to a specific device could be implemented, for example, by storing a device attribute in the certificate's CN (Common Name) field. Verification on the target device would then typically follow these steps:

- Determine the device property
- Read the property stored in the certificate's CN field and compare it with the detected value
- Verify that the certificate itself is valid

Ultimately, this reduces verification to two yes/no comparisons. At the machine level, these become conditional jumps (JZ or JNZ) that even a moderately skilled attacker could patch, allowing the certificate to be reused on another system.

With CodeMeter, by contrast, the private key is derived from the device properties themselves, so patching such checks does not enable a working decryption of the protected application.


Figure 2: Scheme of a CodeMeter Update File

## License Return and Dynamic Data

A key feature of CodeMeter is the secure return of licenses. With a CmDongle or a CmCloudContainer, this process is securely implemented by design.

For a CmActLicense, the license is always replaced with a new one. A continuously increasing generation counter prevents the license file from being reset, and therefore duplicated, after it has been returned. The same mechanism applies to dynamic data such as counters or relative time values.
With token- or certificate-based solutions, a revocation mechanism would need to be implemented instead. In offline scenarios, however, enforcing such revocation procedures is extremely difficult.

## Blockchain

When I was a child, money was simply a number printed on a piece of paper. It could have been a banknote or an entry in a savings book. Today, for most people, money is simply a number stored in a database on a bank's server. But what would happen if someone changed that number to zero? Then all of my savings would be gone "overnight." If you ask a security expert, they might think of a hacker attack. If you ask a skeptic, they might imagine the bank itself doing this on the orders of a government authority. Since I worked in home banking long before joining Wibu-Systems, many years ago, I can say from

personal experience that banks protect their back-end systems extremely well. There are backups, redundant copies, and most likely even backups of those backups.

Even if we dismiss such scenarios as unlikely or unrealistic, there are people who cannot stop thinking about them, especially in the context of the 2008 financial crisis. Under the pseudonym Satoshi Nakamoto, the first blockchain was introduced in 2008 in the white paper "Bitcoin: A Peer-to-Peer Electronic Cash System." The goal was to enable the transfer of digital money without needing to trust a central authority such as a bank. To prevent a digital coin from being spent twice, a central authority normally verifies every transaction. The problem is obvious: you must trust that central authority. Blockchain solves this issue by having all participants collectively maintain the transaction ledger, with transactions cryptographically linked together. Instead of trusting a single entity, the system relies on a consensus mechanism.

If you are wondering what this has to do with software licensing, you are asking exactly the right question. In my opinion: nothing at all. One could argue that blockchain could serve as a tamper-proof ledger for purchases, activations, and similar events. It could theoretically enable the resale of licenses. One might also point to smart contracts and decentralized management. However, in practice, there are several reasons why blockchain is not well suited for software licensing.

### Control

As a software vendor, you want to maintain control over the licenses you issue. This includes use cases such as license revocation, recall, goodwill gestures, and various support scenarios. Blockchain, however, is immutable, so any change must be recorded as a new transaction.

### Offline Systems

Many software applications operate offline, in machines, factories, or isolated networks. Blockchain typically requires network access and a consensus mechanism. This raises the question: what prevents a user from working with an older

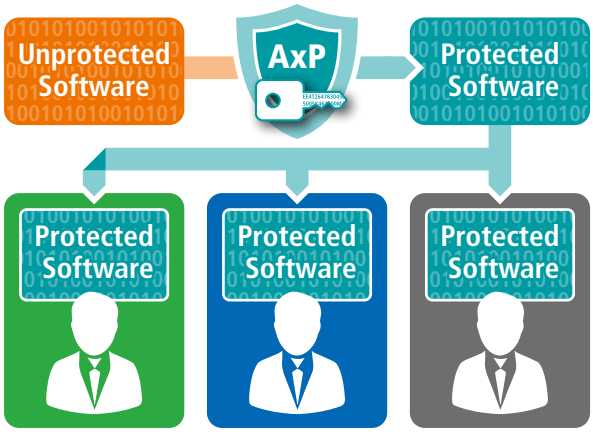



Figure 3: Same software for all users

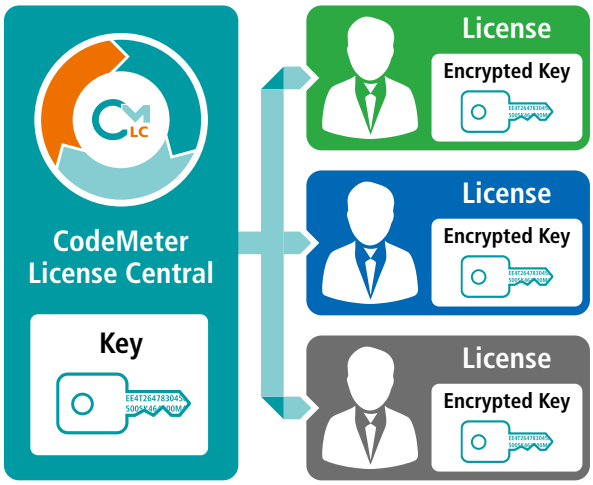



Figure 4: Individual license for each user

copy of the blockchain where the license they want to use has not yet been returned?

### Performance and Costs

Blockchain transactions are slow and can incur fees. This makes them difficult to scale and technically unnecessary for licensing scenarios.

### Identity is Unclear

Blockchain only states: "Wallet A owns license B," without linking that wallet to a verified organization, user, or device. As a result, you would still need identity management, hardware binding, and a licensing server on top of the blockchain. All things considered, blockchain is poorly suited for software licensing. The main reason is its architecture. Blockchain was designed to be trustless, whereas licensing explicitly requires a central authority. Vendors must be able to create, revoke, modify, stop misuse, and resolve support cases. As a software vendor, you must always be able to say whether a license is valid. Blockchain makes this difficult because it is append-only, immutable, and without central control.

CodeMeter already incorporates concepts closely related to certificates and JSON Web Tokens. However, it goes beyond these approaches in several important ways:

- Manipulating the device binding requires far more than patching simple conditional jumps; the private key itself is part of the protection mechanism.
- Secret keys are securely transferred and tightly bound to the target container.
- Licenses can be securely returned.
- Licenses can include usage data such as counters and time-based values.
- With CmDongles and CmCloudContainers, two container types are available whose security exceeds that of CmActLicenses.

# INNO DAYS 2026: Where Security Powers Digital Business

From software protection to AI-driven business models, INNO DAYS 2026 brings together industry leaders, innovators, and researchers to explore how digital technologies are reshaping value creation. Join us in Karlsruhe to learn strategies that turn software security, licensing, and data into sustainable growth.



We are no longer approaching digital transformation. We are in the middle of it. Software has become the core of industrial value creation, while data, connectivity, and artificial intelligence are reshaping how products are built, delivered, and monetized. **What used to be a technical layer is now the core of the business itself.**

INNO DAYS 2026 puts this shift into focus. Hosted at the **Wibu-Systems headquarters in Karlsruhe**, the event brings together key players driving this evolution across industries. On stage, companies such as **CODESYS, Schneider Electric, Siemens Digital Logistics, and TRUMPF** share real-world perspectives on how industrial software is protected, deployed, and monetized. Their insights are complemented by **BSI (the Federal Office for Information Security), Open Source Automation Development Lab, and VDMA**, reflecting the growing alignment between industry, associations, and public institutions.

Academia plays a key role in this dialogue. Contributions from **the Karlsruhe Institute of Technology and Offenburg University of Applied Sciences** bring forward-looking perspectives on communication security and advanced cryptography, bridging cutting-edge research with industrial applications.

A clear pattern emerges: **security is the foundation of digital business**. Protecting intellectual property, controlling how software is used, and ensuring regulatory compliance are becoming prerequisites for market access. At the same time, these capabilities enable new business models, from feature-on-demand to subscription and usage-based licensing.

**Artificial intelligence** accelerates this transformation. It enhances industrial processes, optimizes decision-making, and reshapes customer engagement. But it also raises expectations around trust, integrity, and control. As systems become more dynamic and interconnected, the ability to enforce rules and protect digital assets becomes a competitive advantage.

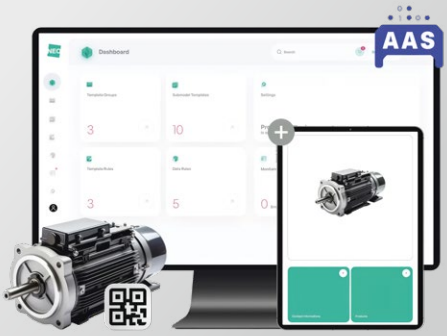
At the infrastructure level, the **Industrial Internet of Things** introduces new complexity. Highly distributed, resource-constrained environments require secure communication across both legacy and modern systems. In parallel, emerging concepts such as confidential computing show how sensitive data can be processed securely, even in shared or untrusted environments, supported by hardware-based trust anchors.

Regulation adds further momentum. With initiatives like **NIS2 and the Cyber Resilience Act**, cybersecurity is becoming enforceable across the entire lifecycle of digital products. This is driving organizations to rethink development, deployment, and long-term maintenance, while also creating opportunities to differentiate through measurable security.

Taking place on **July 15 (afternoon) and July 16**, INNO DAYS is designed for meaningful exchange. Ample time for networking and direct interaction ensures that discussions go beyond presentations. The event will be in English. While the event is designed for in-person participants, a dedicated roundtable will be streamed live on YouTube to extend the conversation.

This is a defining moment. The technologies are here, the frameworks are taking shape, and the competitive landscape is shifting. **INNO DAYS 2026 is where these forces come together and where the next phase of digital business takes shape.**

NEOCEPTION
Pepperl+Fuchs




# Case Study | Neoception

Neoception builds cloud-native software where licensing must be as reliable and scalable as the applications themselves.

## The Challenge

Neoception develops software that needs to run reliably across multiple **Kubernetes** containers. One container runs CodeMeter Runtime and must be reachable and discoverable by the other application pods at all times.

Key challenges included:

- **Automation first:** All required CodeMeter components are stored in Azure Blob Storage and must be retrieved automatically during container image builds.
- **Secure credentials:** Customers provide a .wbc file containing user credentials, which must be injected securely and reproducibly without exposing sensitive data.
- **Dynamic cluster behavior:** License validation must remain functional while pods restart, scale, or are updated.
- **Operational resilience:** Seat-based licenses must remain consistent during restarts and rolling updates to avoid disruptions or unintended restart loops.

## The Solution

Neoception created a dedicated Docker image for CodeMeter with a fully automated build process. All required CodeMeter components, including the runtime and protection tools, are downloaded securely from **Azure Blob Storage** during image creation, ensuring a repeatable and controlled pipeline.

Within Kubernetes, the CodeMeter Container is exposed internally with a stable service identity that application pods can reference through configuration only. This decouples licensing from individual workloads and keeps deployments flexible across environments.

Customer credentials in the form of .wbc files are provided through **Kubernetes Secrets** and never embedded into container images. This ensures security by design while supporting consistent deployments across development, testing, and production.

To guarantee operational stability, Neoception fine-tuned license session behavior so that licenses remain valid during pod restarts, scaling events, and rolling updates. This prevents unnecessary license churn and keeps seat counters accurate even in highly dynamic clusters.

## The Success

With this approach, Neoception achieved exactly what they set out to do:

- **Zero-touch deployment:** Licensing is fully automated and integrates seamlessly into CI/CD pipelines.
- **Resilient operations:** Licensing remains stable through restarts, rollouts, and scaling events.
- **Secure credential handling:** Sensitive assets such as .wbc files are securely managed to ensure compliance.
- **Cloud and on-premises ready:** The solution works consistently in both managed cloud environments and customer-owned infrastructure.

## The Company

Based in Germany, Neoception develops scalable software products that are innovative, practical, and economically viable – fast to implement, easy to operate, and designed to deliver clear business value. The company works closely with customers in manufacturing, logistics, and other industries to build containerized platforms that connect physical processes with digital intelligence.

**Nuno Rodrigues**
**Software Engineer at Neoception**
"Once we had the right setup in place, everything ran as expected. We can now be confident that our customers always run with the right license, without adding operational overhead."

READ MORE

## Join Wibu-Systems at the following events:

**Hannover Messe**
20-24 April 2026
Hanover, Germany
Hall 26, Booth C75

**CyberSec India Expo**
23-24 April 2026
Mumbai, India
Booth C225

**INNO DAYS**
15-16 July 2026
Karlsruhe, Germany

**German Roadshow**
20 October 2026
Hamburg, Germany

**German Roadshow**
27 October 2026
Stuttgart, Germany

**SPS**
24-26 November 2026
Nuremberg, Germany
Hall 6, Booth 428

## Wibu-Systems' INNO DAYS 2026

The premier event for digital strategists aiming to steer their business with a clear, future-focused vision. Attendees will benefit from high-level networking with industry peers, insights from leading experts, and exclusive entertainment. Wibu-Systems' key staff, along with solution partners, clients, and renowned speakers, will lead the discussions. While IP protection and monetization technologies remain central, the event will foster broader engagement on critical industry trends.

**Register asap! The future begins now.**

www.wibu.com/inno-days.html

## Wibu-Systems' Masterclasses

Our masterclasses are designed to share the knowledge and experience behind our technologies and mission, empowering cutting-edge protection techniques, secure software licensing best practices, and versatile monetization models worldwide. Each session combines practical insights from our CodeMeter ecosystem with our commitment to innovation, reliability, and digital trust. Stay tuned for upcoming announcements on our website or through our newsletter, and be ready to register for the sessions that best match your interests.

www.wibu.com/webinars.html

## Get in touch with our local representatives

**WIBU-SYSTEMS USA, Inc.**
USA: +1 800 6 Go Wibu
+1 425 775 6900
**sales@wibu.us**

**WIBU-SYSTEMS (Shanghai) Co., Ltd.**
Shanghai: +86 21 5566 1791
Beijing: +86 10 8296 1560
**info@wibu.com.cn**

**WIBU-SYSTEMS K.K.**
Japan
+81 45 565 9710
**info@wibu.jp**

**WIBU-SYSTEMS Korea Ltd.**
Republic of Korea
+82 2 6206 9490
**sales@wibu.co.kr**

**WIBU-SYSTEMS BV/NV**
The Netherlands: +31 74 750 14 95
Belgium: +32 2 808 6739
**sales@wibu.systems**

**WIBU-SYSTEMS LTD**
United Kingdom | Ireland
+44 20 314 747 27
**sales@wibu.systems**

**WIBU-SYSTEMS sarl**
France
+33 1 86 26 61 29
**sales@wibu.systems**

**WIBU-SYSTEMS**
Spain | Portugal
+34 91 123 0762
**sales@wibu.systems**

**WIBU-SYSTEMS**
Scandinavia | Baltics
+46 8 5250 7048
**sales@wibu.systems**

WIBU SYSTEMS

**SECURITY**
**LICENSING**
**PERFECTION IN PROTECTION**

www.wibu.com


**Imprint**
KEYnote 51 Issue
Spring/Summer 2026

**Publisher**
WIBU-SYSTEMS AG
Zimmerstrasse 5
76137 Karlsruhe, Germany
Tel. +49 721 93172-0
info@wibu.com
www.wibu.com

**Responsible for the content**
Oliver Winzenried

**Editors**
Stefan Bamberg
Marco Blume
Axel Engelmann
Lukas Klassen
Ruediger Kuegler
Daniela Previtali
Andres Prieto
Wolfgang Voelker
Oliver Winzenried

**Design**
Studio Eugen Olchin
Stuttgart, Germany

**Print**
PASSAVIA Druckservice GmbH & Co. KG, Passau, Germany





Picture credits:
Cover: Firefly
Page 4: Firefly
Page 7: Midjourney
Page 10: Midjourney
Page 12: Firefly
Page 14: Midjourney
Page 17: Wibu-Systems
Page 18: Midjourney
Page 22: Midjourney
Page 24: Midjourney
Page 26: Midjourney
Page 31: Neoception

All remaining images are copyrighted by their owner.




5060-002-02 / 2026-03-29