Ponderarium, a place for Cyber Physical System conformity assessment*

Guillaume Nguyen
NADI, University of Namur
guillaume.nguyen@unamur.be

Anthony Simonofski
NADI, University of Namur
anthony.simonofski@unamur.be

Antoine Sacré
NADI, University of Namur
antoine.sacre@unamur.be

Xavier Devroey
NADI, University of Namur
xavier.devroey@unamur.be

Abstract

Nowadays, complex Cyber-Physical Systems (CPSs) are commonly exchanged on the market. However, this complexity does not allow citizens or consumers to properly understand the quality, security, and safety of these products. When considering CPS, such as Advanced Driver Assistance Systems, autopilots on aircraft, or in vitro medical devices, consumers rely on local regulations, international standards, or even simply on their presence on the market to buy, use, and trust these products. Still, when examining regulations and directives provided by the European Union and other governments, only the documentation, not the product, needs to be assessed for compliance. Of course, manufacturers are also interested in knowing if their products satisfy their own set of requirements before putting them on the market. In this paper, we discuss the need for a Conformity Assessment tool, Ponderarium that enables interested parties to assess the quality, security, and safety of CPSs based on static resources. Then, we devise a methodology supported by a first version of Ponderarium that we validate using open-source software for a medical device. The purpose of Ponderarium is to enable the conformity assessment of a CPS from related static resources (such as source code or network frames) with respect to specific requirements extracted from natural language legal texts.

Keywords: Cyber-Physical Systems, Requirements Engineering, Conformity Assessment, IoT, Regulation

* This is the authors' version. The final version is accepted for publication in the *Proceedings of the 59th Hawaii International Conference on System Sciences (HICSS'26)*, Maui, Hawaii, USA.

1. Introduction

The *Mensae Ponderariae* or *Measuring Table* was a place in ancient Roman culture where buyers and sellers could come to measure the volumes and weights of goods exchanged to avoid fraud and altered weighing systems (Cavalieri, 2020). A city magistrate monitored this table to ensure proper measurement. Like the ancient Romans and their care for fairness, the notion of Verification and Validation (V&V) should not rely solely on trusting the provider of a product. In the European Union (EU), regulations and directives (i.e., legal texts) concerning specific products in the context of a European Conformity (CE) marking typically require the production of compliant documentation that describes the quality of the building process. This document needs to be assessed for compliance either by a third party or the manufacturer itself (EC, 2022).

However, since CE markings were mostly focused on physical products, software products were not (at least directly) legally regulated in the EU before 2017 with the arrival of two regulations on medical devices: Regulation (EU) 2017/745 and Regulation (EU) 2017/746 (EU, 2025). In those regulations, software can be considered a medical device under specific conditions. Since then, other legal texts have been published on the *EUR lex* platform such as the Cyber Resilience Act (EU Regulation 2024/2947) on industrial Internet of Things (IoT) and software products, Regulation (EU) 2019/2144 on road vehicles and their systems, Regulation (EU) 2018/1139 on civil aviation and Regulation (EU) 2024/1689 on Artificial Intelligence (AI) (EU, 2025). However, legal compliance is not an easy task. Indeed, Nguyen et al. (2024) show that multiple challenges arise when trying to assess the conformity of a CPS with respect to an EU legal text, such as getting access to required standards

or understanding the technical requirements from such texts. Furthermore, Kempe and Massey (2021) realised a systematic literature review on the presence of academic papers related to Regulatory Compliance (RC) and Security Standards compliance (SSC) throughout the Software Development Lifecycle (SDLC). Their "*findings suggest that academic software engineering research directly connecting regulatory requirements and security standards to later stages of the SDLC is rare despite the industry's focus on regulatory and security standard compliance.*" (Kempe and Massey (2021)). They note that the Software Engineering (SE) community focuses on factors other than RC and SSC.

Cyber-Physical Systems (CPSs) are at the intersection of both physical products and software products, which makes the arrival of CE-related software regulations an even more critical component of their design. Furthermore, the safety and security of users (and bystanders) are directly related to the safety and security of CPSs. If we take a car equipped with an Advanced Driver Assistance System (ADAS) on which the driver is relying, a malfunction or vulnerability of the system could lead to life-critical damages (i.e., a crash). In the EU, the analysis of the (actual) conformity of this ADAS by an accredited body of the EU would only happen after the crash. Indeed, the market surveillance mechanism (supported by multiple legal texts such as Regulation (EU) 2019/1020 (EU, 2025)) prevents further problems while still allowing dangerous products on the market.

Obviously, legal texts from the EU are not the only texts produced to ensure the safety, security, or, more simply, the quality design of products worldwide. Indeed, local regulations, international or regional standards, best practices, etc., all play an important role. From a higher perspective, it relates to producing a human-readable guide or document for interested parties who are willing or required to produce something of a certain level of quality. By all means, tests are typically performed by manufacturers or third parties to ensure the quality of the systems, but not usually by governments or citizen-focused associations. Furthermore, the compliance assessment often relies on the documentation of the building process (V&V tests might be a part of what is required in a standard, but those are usually not part of the compliance assessment).

Nonetheless, Requirements Engineering (RE) delivers powerful techniques to formalize the requirements of a system from a human-written text. However, verifying and validating compliance with such requirements requires a higher level of abstraction of the System Under Conformity Assessment (SUCA), such as models for Model-Based Testing (MBT) or documentation of the SUCA.

In this paper, we propose *Ponderarium Animae Machinarum* or *The tool to weigh the soul of machines*, which will later be referred to as *Ponderarium*. On one plate of the scale, we place computer code or any other static resources, such as network frame captures. On the other plate, we place technology-specific *signals* related to the SUCA, based on specific documents (such as legal texts, standards, internal documents, etc.). Signals are potential elements of interest, such as access to specific communication channels or data protection mechanisms, required by a legal text, standard, or other relevant document. This leads us to our first research question: **RQ1.** *In what ways can the conformance of a CPS to a reference text be evaluated using related static resources, while ensuring traceability between them?* To answer that question, Ponderarium will produce a human-readable report with all signals found to let the user assess the quality of the SUCA. Among the different analysis techniques, static analysis will be used, leading to a sub-research question: **RQ1.1.** *In what ways does leveraging technology-specific static analysis alleviate the effort involved in assessing the compliance of a CPS?*

Our primary goal is to maintain traceability between each signal and each requirement, which is one of Ponderarium's goals. However, to be able to identify signals, legal texts and regulations should be detailed enough, leading to our second research question: **RQ2.** *To what extent do legal texts and related documents provide a sufficient level of technical detail to support a meaningful static resource analysis?*

2. Background and related work

In the following sections, we highlight the need for our approach by explaining what we define as CPS (Section 2.1). Then we put the emphasis on RE and what it means in the context of legal requirements (Section 2.2). Finally, we explore scientific research related to compliance assessment and the various related methods.

2.1. Cyber-Physical Systems

CPSs are a broad category of systems. Indeed, it includes concepts such as Embedded Systems (Lee and Seshia, 2017), IoT, Real-Time Computing, and even Operational Technology (OT). In this paper, we rely on the definition from Rajkumar et al. (2010): "*Cyber-physical systems (CPS) are physical and engineered systems whose operations are monitored, coordinated, controlled, and integrated by a computing and communication core.*" Rajkumar et al., 2010, p. 1

Furthermore, Tekinerdogan et al. (2020) define ten

application domains: Health, Smart Manufacturing, Transportation, Process Control, Defence, Building Automation, Robotic Services, Critical Infrastructure, Emergency Response, and Other. Those domains show the omnipresence of CPSs in society, thus the critical importance of their safety and security compliance with industry standards and applicable regulations. They define a metamodel to represent a CPS based (among other things) on the domain of application and the various components. Similarly, Walch and Karagiannis (2019) proposed including design thinking when modeling CPSs. Their approach consists of a three-layer s*IoT (or s-star IoT) architecture allowing the building of CPS based on requirements extracted from user concerns. Furthermore, Model-Driven Engineering (MDE) provides a strong foundation for designing CPS using other modeling languages, such as UML, MARTE, and SYSML (Gonçalves et al., 2017). Those complex models tackle the intricate relations between real-time constraints and computing capabilities. Concerning the development of CPS, Jakobs et al. (2019) proposed an approach to conceive dynamic and trustworthy CPS by specifying software components and related functional requirements using TLA+, a formal language.

2.2. Requirements Engineering

Kotonya and Sommerville (1998) define RE as the collection of *processes involved in developing system requirements* (p.6) and system requirements as *what the system is required to do and the circumstances under which it is required to operate* (p.3). Although this definition comes from the 1990s, the concept itself finds roots as early as the 1960s for military project management, such as the *Minuteman ground electronics system for WS-133B* (Dresner and Borchers, 1964). What we keep from RE concepts here is the need to understand what is required of a system when designing it and how to check if the requirements are satisfied. When focusing on legal texts, the financial impact of a non-compliant system can be as significant as that of a failed functional device. Indeed, recalled or banned products can no longer be sold on a specific market through mechanisms such as the EU *Market Surveillance* with the *Safety Gate* online platform for public consultation (EC, 2025). Of course, the main purpose of product-related regulations is to ensure the safety and security of citizens, which is why, on the one hand, governments should understand the needs of the industry, but also consider the societal implications of technologies allowed on the market. On the other hand, manufacturers should easily understand what is required of them and not only produce compliant documentation as if it were an administrative constraint, but truly offer safe and secure products.

In the field of legal compliance, RE automation has been a concern for quite some time now (Kiyavitskaya et al., 2008). Aberkane et al. (2021) carried out a Systematic mapping study in RE for the General Data Protection Regulation (GDPR) (EU Regulation 2016/679) (EU, 2025). They identified commonalities between RE and Natural Language Processing (NLP) in general, as well as opportunities to bridge the gap between NLP and GDPR in RE, given that NLP techniques offer potential for automating manual RE tasks. Sleimi et al. (2021) analyzed the current landscape of RE and NLP in the context of legal texts. They showed that this research area lacked harmonization and coverage in terms of legal metadata. They reviewed the available literature and proposed an approach for automating the collection of legal metadata using NLP and Machine Learning (ML) techniques.

Recent advances in AI, and more specifically, Large Language Models (LLMs), have revealed a significant opportunity in RE, as well as in terms of compliance assessment. Examining system and software development further, Hassani et al. (2025) utilize LLMs to extract software and systems-related requirements from Food Safety Regulations. They manage to provide a pipeline providing an accurate (up to 90% precision) classification of requirements in multiple food-safety regulations after manual validation. Interestingly, they identified similar software and system-related concerns to those reported by Amaral et al. (2022), specifically focusing on extracting information from privacy policies.

On a higher level, Guber and Reinhartz-Berger (2024) provide an SLR on the reuse of compliant software in terms of privacy during the early phases of development. They show that while interest in this topic is on the rise, there are still many challenges left when producing a privacy-compliant software, even when reusing compliant artifacts, as well as understanding the requirements of various privacy-related regulations.

2.3. Compliance Assessment

Compliance assessment is a crucial step in ensuring a product conforms to a specification. Many approaches are available to perform such an assessment. Indeed, Torre et al. (2019) developed a model for the EU General Data Protection Regulation (GDPR) to facilitate automatic compliance checking. Their approach begins by creating a generic model based solely on the initial GDPR text. Then, they tailor it to available

materials from the EU, as well as domain-specific requirements. Finally, they suggest a way to instantiate the domain-specific model with a real-life application and check the conformity of the SUCA. Similarly, Sacre et al. (2021) suggest creating two models separately, one representing a GDPR-compliant implementation of a SUCA and one representing the actual SUCA. Then, they check the differences between the two models and assess the conformity of a SUCA based on the similarities between the models.

Abualhaija et al. (2025) use LLMs to retrieve compliance information using a question-and-answer approach. They also suggest a method for automated compliance checking using RE, NLP, and ML techniques to extract requirements from legal texts and identify similarities in SUCA documentation. Then, they assess whether the statement in the documentation complies with the target regulation (e.g., if the documentation includes a point about data breaches, the tool identifies similarities with GDPR data breach requirements and evaluates the compliance of the statement, such as the delay before notifying the data protection authority).

Concerning methods for checking compliance during the design phase, Kaneen and Petrakis (2020) provide a methodology based on UML to provide a compliant IoT application. They ask 10 questions to be answered for the creation of a complete class diagram, which is then queried for compliance assessment. However, they recognize that they might overlook the whole complexity of the GDPR. Shifting focus on the EU AI act (Regulation (EU) 2024/1689), Kelly et al., 2024 base their approach on *design contracts* to define guarantees that a set of requirements is satisfied. They developed a high-quality model for AI that bridges the gap between the requirements of the act and technical requirements. On a more organizational point of view, de la Vara et al. (2025) focus on standards such as ISO 60304 on medical device software. This standard outlines techniques that should be in place throughout the Software Development Lifecycle. Their approach is based on Knowledge Centric Systems Engineering (KCSE). They provide a way to verify and validate the different requirements for SES (a commercial KCSE) used by large companies. They performed system artifact quality analysis while maintaining traceability to the original requirements.

Nevertheless, those techniques rely on creating a higher-level representation of a SUCA. Fortunately, interesting work has been done to assess the conformity of a system directly. Bicaku et al. (2020) provide us with a complex tool for the continuous compliance assessment of industrial CPS. By extracting *Measurable Indicator Points* (MIPs) from international standards, they develop technology-specific extraction modules that query industrial CPS directly to verify their correct configuration during production. That means the conformity of a SUCA can be assessed almost in real-time, and each *test* can be traced back to a requirement. They also suggest that multiple standards can have similar requirements and that a single module (or test) could be used to tick multiple compliance boxes (when multiple standards are relevant). In the context of EU-related regulations, *Harmonized Standards* are relevant as they serve as an instrument to help manufacturers understand what is required of them and how to comply with regulations (Nguyen et al., 2024). Nguyen and Devroey (2025) built a tool to assess the conformity of an Android Application as a *Medical Device* based on the international standard IEC 62443 on securing and maintaining industrial automation and control systems (IACS). They performed a static analysis to assess whether or not *signals* are present within the code of an application and returned a *.csv* file with results to be visualized.

3. Ponderarium

In this section, we explain the high-level approach of Ponderarium as represented in Fig. 1. We show that the main idea is to extract requirements from a natural text and compare them to data retrieved from a SUCA. For example, we have the following process: **(1) Text:** *The medical device regulation*, **(2) Requirements:** *Produce a qualitative device*, **(3) Technical Requirements:** *The temperature measurements are expressed in Celsius degrees*. Those requirements are then filtered to retain the technological requirements that can be assessed by examining the computer code or other static resources via *signals*. **(4) Technology-specific:** *We express the use of a temperature sensor with Celsius degrees in a specific programming language*. Then, **(5) Ponderarium:** *compares the Technology-specific requirements with the data retrieved from the SUCA* and **(6) Produce the assessment results**. The important part of such a process is to maintain traceability from the initial requirements extracted from text to the assessed signals in the SUCA. We define a *signal* as neither good nor bad but as a piece of information that an assessor could use to interpret the conformity of a SUCA to a specific requirement. Steps (5) and (6) represent the main contributions of this paper.

On the other scale of Ponderarium, we put the data from the SUCA in and let our tool look for the presence of *signals* related to the assessed requirements. Specifically, we perform static analysis by grepping (i.e.,

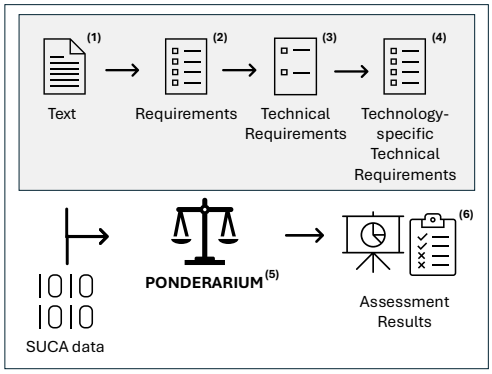

Figure 1. The high-level architecture of Ponderarium.

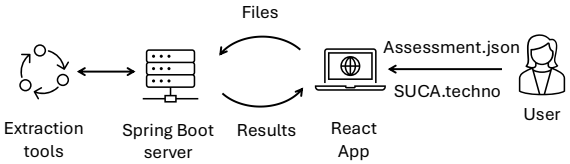

Figure 2. The low level architecture of Ponderarium.

global regular expression print or grep) particular strings of characters, as well as using regular expressions from computer code (or any other resources) which return a `true` or a `false` value, indicating whether Ponderarium found the signal or not. For example, we want to look for '*#include <SoftwareSerial.h>*' in a computer code from an Arduino project. Our tool will read each line and, whenever it finds a match with the text, return `true` as the result of the signal.

Finally, we use preconditions to run an analysis. Indeed, a requirement could be related to multiple signals, and signals could be related to other signals. For example if we want to assess the configuration of an LCD screen such as the refresh rate (e.g., 10Hz) and the clarity of the information displayed (e.g., a heart monitor displaying *BPM* for Beat Per Minute next to its value), we also need to make sure that the computer code calls LCD screens for displaying its information to the user. If not, then the assessment could be considered a failure, even if it is irrelevant to our assessment. Those preconditions differ from the preconditions mentioned in legal texts or standards (or others); they are solely used to determine if an assessment should be carried out. Of course, the SUCA should only be assessed in accordance with the relevant article from the legal texts (or other sources).

Performing static analysis using string matching and regular expressions facilitates the definition of signals, albeit at the expense of potentially losing information from the SUCA. Experimenting with other, more complex static analysis approaches (e.g., CodeQL) and their applicability for non-static analysis experts (i.e., developers, testers, and requirements engineers) is part of our future work. Nevertheless, following work from the Android community and more specifically Arzt et al. (2014)'s on *Flowdroid*, we can see that they manage to classify benign and malicious Android applications based on the *Manifest* file. This small file within every Android application includes the *permissions* required by an application to perform its tasks. When a permission does not suit the purpose of the application, Flowdroid triggers a warning (e.g., granting an SMS permission for a mobile game). Thus, this type of simple analysis can be leveraged to assess the presence of good or bad compliance signals within static resources, such as computer code.

4. Implementation

In this section, we will explain how we built Ponderarium. Figure 2 shows the view of the various technologies involved in the assessment process. On the right side, you can see a user providing two files, one with the assessment data (the definition of signals to be collected) as a JSON file and the other with a file related to the SUCA (computer code, etc.). Then the web application (React) transfers it to the Spring Boot (Java) server, which extracts information from the SUCA file and collects the signals. The results are provided to the web application and displayed to the user as a dashboard. The architecture of Ponderarium is designed to alleviate the user from the computational requirements. Furthermore, a modular approach allows for more complex capabilities and flexible upgrades such as de-compiling code (e.g., Android Applications) or AI-powered analysis and result explanation.

The *json* file structure is composed of: (1) the *article* field used to keep the traceability to the original requirement; (2) the *assessment* field with a boolean expression to express complex conditions over the results of related *signals* (e.g, ***sig01*** *AND NOT* ***sig02***); (3) an optional *pre-condition* field to store the condition for an assessment to be carried out.

Algorithm 1 shows the full evaluation process. For each assessment, Ponderarium will evaluate the precondition using the signals provided as input (line 4). The $evaluate(expr, S)$ function will compute the different signals by evaluating the signals appearing in $expr$. To avoid recomputing the same signals multiple times, S serves as a memory to store the results of already evaluated signals. If the precondition is respected, Ponderarium proceeds to the evaluation of the

Algorithm 1: Ponderarium evaluation process

Input: S: set of known signals $(sig\_id, boolean)$, A: set of assessments to evaluate, C: source code
Result: S: updated set of signals, A: set of evaluated assessments

1 Remove comments and empty lines from C;
2 **foreach** $a \in A$ **do**
3 $p \leftarrow$ precondition of a;
4 $evaluate(p, S)$;
5 **if** *p is empty* ***or*** *p is* `true` **then**
6 $result \leftarrow evaluate(a, S)$;
7 **if** *$result$ is* `true` **then**
8 Mark a as `PASSED`;
9 **else**
10 Mark a as `FAILED`;
11 **else**
12 Mark a as `NA`;
13 **return** S, A;

current assessment a (line 6). Depending on the results, a is marked as passed or failed. If the precondition is not respected, the assessment is skipped and marked as `NA`. To assess the different boolean expressions, we rely a language parser (ANTLR4) to generate a *visitor* with a grammar allowing the use of *OR*, *AND*, and *NOT* operators as well as parentheses for more complex expressions handling.

The manual part in this process is still extensive. The user needs to produce a set of signals to be assessed. Such a process ensures the traceability between produced signals and the requirements extracted from a text. The extraction of the data in the SUCA file is automated. However, we need to create modules for each technology under assessment. To ensure the scalability of our tool, we created *collector* and *assessor* interfaces, which need to be implemented for each technology. The collector is used to collect, preprocess, and formalize data from the SUCA file, and the assessor performs the conformity assessment by matching signals with the provided data from the SUCA. In the case of computer code, the collector removes all comments to minimize the risk of false positives in the results. Then, the data is transferred to the assessor, which performs the actual assessment using the grep and regular expressions of the different signals. Currently, we can work with Arduino code (INO) files and Android applications (APK) files.

Ponderarium is designed and built to allow future evolutions. The goal is to allow it to grow to handle other technologies for SUCA assessment as well as the creation of requirement-based signal collectors from texts written in natural languages. Thus, using a back-end for data processing is necessary in the long run.

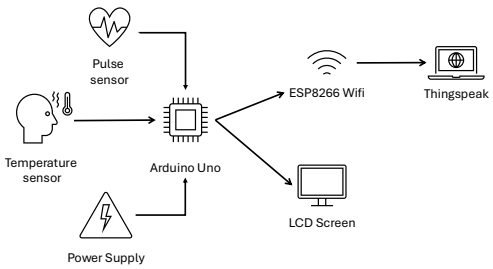

Figure 3. The overview of the components of the SUCA (ElectronicsWorkshop, 2023).

5. Evaluation

In this Section, we will answer our research questions by evaluating Ponderarium qualitatively. We will perform the conformity assessment of a health-related Arduino project called *Patient Health Monitoring Based On IoT using ESP8266 & Arduino*(ElectronicsWorkshop, 2023). This CPS, as presented in Figure 3, is not CE certified, and, being a healthcare system, it must comply with several regulations, making it an ideal example for this initial evaluation. The project is composed of:

- Arduino Uno (ATmega328P): A simple microprocessor with a recommended power supply of 7-12V.
- Sensors: Temperature and Heartbeat.
- Actuators: LCD Screen and ESP32 Wi-Fi.
- Connection: data sent to thingspeak.com.

Its purpose is to monitor patient data using an LCD screen and also send data online to thingspeak.com using a specific API key. This website enables IoT projects to send data for online visualization (charts, dashboards, etc.), facilitating remote monitoring. Based on the documentation, we can define the device as a *Class Im* medical device or a Class I medical device for measuring purposes, as the medical purpose is clearly stated. If someone wanted to use this project without qualifying it as a medical device, one should only mention that it is not built for medical purposes (for purely legal reasons).

Our methodology for the assessment of this project will be the following: (1) Use the Medical Device Regulation (MDR) or Regulation (EU) 2017/745 to perform a conformity check; (2) select legal articles and parts of the MDR that relate to the use of technology; (3) translate the articles into technological requirements; (4) create a signals collector for requirements relevant to the project; (5) assess the project using Ponderarium; (6) display the results of the assessment; (7) conduct a security and safety gap analysis of the project compared

to the assessed requirements; and (8) summarize the lessons learned.

5.1. Analysing the Medical Device Regulation

First, we can see from reading the MDR that the focus is on *Quality* from an ISO perspective: the main concern of this regulation is to ensure that manufacturers have effective control over their company and implement good quality management, risk management, and other relevant systems. However, Article 5 of the MDR (*Placing on the market and putting into service*) states that " *2. A device shall meet the general safety and performance requirements set out in Annex I, which apply to it, taking into account its intended purpose.*" (European Parliament and Council of the European Union, 2017) Which directly relates to the device and not the manufacturer. When examining Annex I of the same document, we can identify multiple points that may be related to technological features. We kept the following articles from the annexes: articles. 4, 5, 6, 14, 15, 16, 17, 18, 19, 21, and 22 of Annex I and article 6 of Annex VI. For each article, we provide a brief description of its content from a technological perspective. Here is an example of Article 18. Please refer to the original article for any legal application:

Annex I art. 18 *on active devices and devices connected to them. They shall be robust against single fault condition, embed a mechanism to see the remaining battery level (when relevant), embed alarms for power supply failure and medical monitoring (e.g., life-critical readings), have electromagnetic robustness and limit its own emission, avoid electrical shocks. Devices shall be designed and manufactured in such a way as to protect, as far as possible, against unauthorised access that could hamper the device from functioning as intended.*

5.2. Selecting relevant articles and creating static analysis assessments

Concerning our SUCA, we selected the following articles. 4, 5, 6 (captor fault detection), 14, 15, and 18 of Annex I and article 6 of Annex VI. The first author, who has a background in computer science, management, and certification, created signals based on available documentation and online research. The second author, who has a background in computer science and law with a focus on conformity assessment, also performed a relevant article selection based on the SUCA. Then, both authors discussed and validated their choices together. For example if we look at a specific requirement for *Annex I art. 18*, the point 4 states that *Devices intended to monitor one or more clinical parameters of a patient shall be equipped with appropriate alarm systems to alert the user of situations which could lead to death or severe deterioration of the patient's state of health.*. As we can see, this applies to our SUCA. We can derive the following technological requirements: *Alarm systems for patient heart-rate conditions warning* and *Alarm systems for patient temperature conditions warning*. When looking at those two technological requirements, we can elaborate, for example, the following *assessment*:

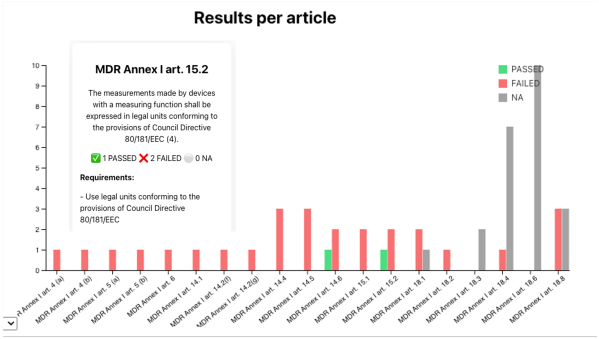

Figure 4. Excerpt of PonderariumDashboard page.

identifier	ALARM-HEART-01
description	Verifies that an alarm is raised if heart rate exceeds an upper limit.
requirement	War-He-Heart
pre-condition	SIG_ReadHeartRate
assessment	SIG_HighHeartRateCondition AND SIG_TriggerAlarm

Traceability is maintained through the *requirement* field. Indeed, elaborating technological requirements from a legal requirement requires a deeper understanding of the technological implications. The regex of SIG_ReadHeartRate is '`readHeartRate\\s*\\(\\)`'. However, if there is no *readHeartRate* function implemented, it doesn't mean that the SUCA is not performing any heart-rate measurement. It could also mean that it uses an alternative naming convention or a different method of measuring a patient's heart rate, so there is a need to have multiple possible signals to cover as many possible implementations as necessary. As our SUCA only monitors heart rate and temperature conditions, there is no need to assess any other conditions. If we extended the context of this assessment, more requirements would be related to Annex I article 18.4. Furthermore, we did not find any European Standard (EN) relevant to the analysis of our SUCA based on the list of harmonized European Standards (hEN) provided with the MDR.

5.3. Performing the assessment and analyzing the results

Now that we have selected the relevant articles, created technological requirements and sets of *assessments* (52 in total) with related signals, we can start the assessment using Ponderarium. Figure 4 shows the final results of the assessment. From the 52 assessments, 2 were *PASSED*, 27 were *FAILED*, and 23 were not applicable (due to the signal in the pre-condition not found). We can see that there was a use of an LCD screen that was detected for the ergonomic function of the device (MDR Annex I art. 14.6) and that BPM were used. At the same time, temperature was measured in °F and not °C (MDR Annex I art. 15.2). By comparing those results with our manual analysis, we do partially agree with it, as we found other positive signals concerning the measuring of temperature and heart-rate data that could not be assessed with our assessments. Indeed, the static analyzer didn't allow for creating a pertinent signal, as knowledge gets lost when variables are used to store information. Detecting such signals requires more advanced analysis, such as dataflow analysis. More interestingly, we found concerning signals regarding Annex I art. 4(b) on *protection measures of each hazard, conform to safety principles, taking account of the generally acknowledged state of the art*. The fact that the device sends health data directly to an uncontrolled web server with a single 16-character *String* as security token could pose serious security issues (data breaches, etc.). Nevertheless, this requirement is vague in the regulation and could be subject to interpretation. Indeed, *State of the art* should directly refer to a European or international standard.

6. Discussion

6.1. Conformity Assessment (RQ1.)

Currently, we utilize static analysis through grep and regular expressions to examine the source code. Although simple, it has the advantage of being easily used and understood by non-static analysis experts. Our evaluation demonstrates the relevance and applicability of this approach, with 27 out of 52 *assessments* marked as *FAILED* (RQ1.1.). Our manual analysis also demonstrated that we had almost found all the observations from the manual audit. Indeed, more advanced techniques such as dataflow analysis and tools such as CodeQL are required to enable the detection of the signals missing compared to the manual analysis. Circling back to Section 2, we proposed a methodology that could be applied to any CPS technology, given we have static resources available without the need for the creation of a higher abstraction representation (such as a model), which was not yet proposed in the reviewed articles. Indeed, our work improves a previous approach presented in Nguyen and Devroey (2025) by leveraging a mechanism that can be adapted to the technology of a SUCA, similar to Bicaku et al. (2020), and we do not require the creation of a SUCA-specific model like Kaneen and Petrakis (2020).

Furthermore, while our current static analysis approach is limited as we cannot cover every possible implementation of a medical device for a single technology, it offers advantages compared to traditional industrial assessment techniques (validating analog input with expected digital output), in particular for *cyber-security* related requirements. For example, sending controlled pulse signals to the device and verifying that the BPM output readings match the input could be validated, while we would not see that the BPM readings are sent to an external web server. In other words, a perfectly built device could send patient data to remote locations without being detected by the current analysis used by Ponderarium.

6.2. Regulations Technical Details (RQ2.)

Concerning the technical level of legal texts, we conclude that the level was subject to too much interpretation to be specific enough (**RQ2**). The fact that we did not find a relevant European Standard in the list of harmonized European Standards also shows the lack of specificity concerning software-related requirements in the MDR. In particular, this means that the assessment effort still requires manual work to refine requirements and derive related *assessments* and *signals*. In the long term, a set of assessments related to a specific technology could be reused for every SUCA using the same technology (e.g., Arduino).

6.3. Limitations and Future Work

Regarding Ponderarium itself, besides the previously discussed use of more advanced analysis techniques for signal detection, we envision utilizing other artifacts, such as system logs and test execution results, to further enhance the capabilities of Ponderarium. Secondly, as emphasized by RQ2, the requirements engineering process described in Fig. 1 entails a significant amount of manual work. We will explore how it can be automated using machine learning techniques (e.g., Retrieval-Augmented Generation), following the approach of other researchers, while keeping the traceability to the original textual requirements.

Regarding our evaluation, we analyzed the results

ourselves. This could be explained by the very specific set of skills required by potential users of our approach: conformity assessment combines computer science and legal skills, which is rare to find in a single individual to set up a full-scale user experience. For this initial evaluation, we chose to rely on the expertise of the first two authors, encompassing legal, compliance assessment, and software engineering. A more thorough evaluation with external users is part of our future work. Similarly, we used only one project due to the amount of manual effort required to gather legal requirements and translate them into technical requirements and associated signals. However, this paper presents an approach to a methodology that can be applied to various technologies related to CPS. Experimenting with other types of CPSs is also part of our future work.

7. Conclusion

In this paper, we developed Ponderarium, a technology-specific static analysis conformity assessment tool. Although the current static analysis technique is limited, our evaluation still demonstrated the relevance of our approach for conformity assessment of medical device software. In particular, almost all the observations from the manual audit were also identified using Ponderarium, and signals of violations could be traced back to technology-agnostic legal requirements. The low technical level of legal texts leaves space for too much interpretation, entailing a significant manual effort to derive corresponding signals, but also emphasizing the importance of the traceability capabilities offered by Ponderarium.

While this work has some limitations, it presents multiple encouraging opportunities in the domain of compliance assessment of CPS through static resource analysis.

Acknoledgements. This research was funded by the CyberExcellence by DigitalWallonia project (No. 2110186), funded by the Public Service of Wallonia (SPW Recherche).

References

Aberkane, A.-J., Poels, G., & Broucke, S. V. (2021). Exploring automated gdpr-compliance in requirements engineering: A systematic mapping study. *IEEE Access*, *9*, 66542–66559.

Abualhaija, S., Ceci, M., & Briand, L. (2025). Legal requirements analysis: A regulatory compliance perspective. In *Handbook on natural language processing for requirements engineering* (pp. 209–242). Springer. https://doi.org/10.1007/978-3-031-73143-3_8

Amaral, O., Abualhaija, S., Torre, D., Sabetzadeh, M., & Briand, L. C. (2022). Ai-enabled automation for completeness checking of privacy policies. *IEEE Transactions on Software Engineering*, *48*(11), 4647–4674.

Arzt, S., Rasthofer, S., Fritz, C., Bodden, E., Bartel, A., Klein, J., Le Traon, Y., Octeau, D., & McDaniel, P. (2014). Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for android apps. *ACM SIGPLAN Notices*, *49*(6), 259–269.

Bicaku, A., Tauber, M., & Delsing, J. (2020). Security standard compliance and continuous verification for industrial internet of things. *International Journal of Distributed Sensor Networks*, *16*(6), 155014772092273.

Cavalieri, M. (2020). Les villes « à la romaine ». In R. G. Villaescusa, G. Traina, & J.-P. Vallat (Eds.), *Les mondes romains. questions d'archéologie et histoire* (pp. 197–222). Ellipses Édition.

de la Vara, J. L., Morote, J. M., Ayora, C., Giachetti, G., Alonso, L., Mendieta, R., Muñoz, D., Nolasco, R. R., & González, A. (2025). Early v&v in knowledge-centric systems engineering: Advances and benefits in practice. *Proceedings of the 2025 IEEE Conference on Software Testing, Verification and Validation (ICST)*, 498–509.

Dresner, J., & Borchers, K. H. (1964). Maintenance, maintainability, and system requirements engineering. *SAE Technical Paper Series*.

EC. (2022). Commission notice the 'blue guide' on the implementation of eu product rules 2022 [[Accessed 10-10-2024]]. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=uriserv:OJ.C_.2022.247.01.0001.01.ENG

EC. (2025). Safety Gate: The EU rapid alert system for dangerous non-food products [[Accessed 09-05-2025]]. https://ec.europa.eu/safety-gate/#/screen/home

ElectronicsWorkshop. (2023). Patient Health Monitoring Based On IOT using ESP8266 & Arduino [[Accessed 31-05-2025]]. https://electronicsworkshops.com/2022/11/14/patient-health-monitoring-using-esp8266-arduino/

EU. (2025). EUR-Lex Access to European Union law [[Accessed 31-05-2025]]. https://eur-lex.europa.eu/

European Parliament and Council of the European Union. (2017, April). Regulation (eu) 2017/745 of the european parliament and of the council of 5 april 2017 on medical devices, amending directive 2001/83/ec, regulation (ec) no 178/2002 and regulation (ec) no 1223/2009 and repealing council directives 90/385/eec and 93/42/eec [Official Journal of the European Union, L117, pp. 1–175].

Gonçalves, F., Rettberg, A., Pereira, C., & Soares, M. (2017). A model-based engineering methodology for requirements and formal design of embedded and real-time systems. *Proceedings of the 50th Hawaii International Conference on System Sciences (2017)*.

Guber, J., & Reinhartz-Berger, I. (2024). Privacy-compliant software reuse in early development phases: A systematic literature review. *Information and Software Technology*, *167*, 107351.

Hassani, S., Sabetzadeh, M., & Amyot, D. (2025). An empirical study on llm-based classification of requirements-related provisions in food-safety regulations. *Empirical Software Engineering*, *30*(3).

Jakobs, C., Werner, M., & Tröger, P. (2019). Dynamic composition of cyber-physical systems. In T. Bui (Ed.), *52nd hawaii international conference on system sciences, hicss 2019, grand wailea, maui, hawaii, usa, january 8-11, 2019* (pp. 1–10). ScholarSpace / AIS Electronic Library (AISeL).

Kaneen, C. K., & Petrakis, E. G. (2020). Towards evaluating gdpr compliance in iot applications. *Procedia Computer Science*, *176*, 2989–2998. https://doi.org/10.1016/j.procs.2020.09.204

Kelly, J., Zafar, S. A., Heidemann, L., Zacchi, J.-V., Espinoza, D., & Mata, N. (2024). Navigating the eu ai act: A methodological approach to compliance for safety-critical products. *2024 IEEE Conference on Artificial Intelligence (CAI)*, 979–984. https://doi.org/10.1109/cai59869.2024.00179

Kempe, E., & Massey, A. (2021). Regulatory and security standard compliance throughout the software development lifecycle. *Proceedings of the 54th Hawaii International Conference on System Sciences*.

Kiyavitskaya, N., Zeni, N., Breaux, T. D., Antón, A. I., Cordy, J. R., Mich, L., & Mylopoulos, J. (2008). Automating the extraction of rights and obligations for regulatory compliance. In *Conceptual modeling - er 2008* (pp. 154–168). Springer. https://doi.org/10.1007/978-3-540-87877-3_13

Kotonya, G., & Sommerville, I. (1998, April). *Requirements engineering*. John Wiley & Sons.

Lee, E. A., & Seshia, S. A. (2017, December). *Introduction to embedded systems* (2.2). MIT Press.

Nguyen, G., & Devroey, X. (2025). A3S3 - automated android audit of safety and security signals. In L. Pufahl, K. Rosenthal, S. España, & S. Nurcan (Eds.), *Intelligent Information Systems - CAiSE 2025* (pp. 205–212, Vol. 557). Springer. https://doi.org/10.1007/978-3-031-94590-8_25

Nguyen, G., Knockaert, M., Lognoul, M., & Devroey, X. (2024). Towards comprehensive legislative requirements for cyber physical systems testing in the european union. https://doi.org/10.48550/ARXIV.2412.04132

Rajkumar, R., Lee, I., Sha, L., & Stankovic, J. (2010). Cyber-physical systems: The next computing revolution. *Proceedings of the 47th Design Automation Conference*.

Sacre, A., Colin, J.-N., & Hosselet, B. (2021). Arrcis: Évaluation et renforcement de la conformité réglementaire d'un système d'information. In *Time to reshape the digital society* (pp. 159–176). Larcier.

Sleimi, A., Sannier, N., Sabetzadeh, M., Briand, L., Ceci, M., & Dann, J. (2021). An automated framework for the extraction of semantic legal metadata from legal texts. *Empirical Software Engineering*, *26*(3). https://doi.org/10.1007/s10664-020-09933-5

Tekinerdogan, B., Blouin, D., Vangheluwe, H., Goulão, M., Carreira, P., & Amaral, V. (2020, November). *Multi-Paradigm modelling approaches for cyber-physical systems*. Academic Press.

Torre, D., Soltana, G., Sabetzadeh, M., Briand, L. C., Auffinger, Y., & Goes, P. (2019). Using models to enable compliance checking against the gdpr: An experience report. *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS)*. https://doi.org/10.1109/models.2019.00-20

Walch, M., & Karagiannis, D. (2019). How to connect design thinking and cyber-physical systems: The s*iot conceptual modelling approach. *Proceedings of the 52nd Hawaii International Conference on System Sciences*.