

"FROM ART TO ALGORITHM" – FORMALIZATION AND MODELING OF LEGISLATIVE PROCESSES WITH A FOCUS ON HEALTHCARE

Adam Vojtěch, Vojtěch Merunka, Jiří Kofránek

Abstract

This article deals with the formalization and modeling of legislative processes in healthcare. It emphasizes the importance of cooperation between architects, implementers, and users of information systems from the very beginning of their development. It highlights the pitfalls of creating information systems for public administration, which require the incorporation of a precise description of the processes used in legislation. Using the example of our own experience with solving the issue of ePrescription and creating legislation for the digitization of vaccination records, it shows the importance of using visual simulation methods and tools to facilitate mutual understanding between healthcare professionals, legislators, and software developers, for early detection of errors, and for significantly shortening the legislative process. The article also discusses current methods of formalizing legislative drafting. It describes a promising, internationally standardized object-process methodology (OPM) and our experience with the OPCloud tool that implements it. In conclusion, recommendations are formulated for the future strategy and implementation of formalization approaches that facilitate interdisciplinary collaboration and the development of hybrid approaches combining formal process analysis methods with artificial intelligence language models.

Keywords

AI, formalization, modeling, language models, legislation, eHealth

1 Introduction: From Cathedrals to eHealth

1.1 Why we build digital worlds differently than Parlěj built St. Vitus Cathedral

Medieval masters, such as Petr Parlěj during the construction of St. Vitus Cathedral, embodied a dual profession: they were both designers and implementers. Their complex and 'esoteric' know-how was passed down from generation to generation and laid the foundations for modern engineering. It is from the brotherhoods of these skilled builders and stonemasons, associated in building huts – construction and stonemasonry workshops, called "loges" in France – that the name and origin of Masonic lodges is derived.

With the development of technical fields, the profession logically divided into **architects** and **builders**. Architects design buildings in terms of their appearance and the purpose they are to serve. They therefore primarily address the question **of WHAT** the user wants from the future building. In their designs, they combine aesthetic and construction knowledge. They plan the internal and external layout of the building so that it best meets the needs of future users. They design shapes and materials with a view to ensuring that the building is technically feasible. Civil engineers then address the question **of HOW** to implement the building according to the architectural design and how to ensure its stability and safety. They manage the gradual implementation and are responsible for ensuring that the resulting building is technically robust and corresponds to the architect's original intention.

A similar division between **architects** and **implementers** also appears in the creation **of information systems**. This is because it is not just about programming the software itself, but above all about designing solutions for users' needs and describing how the system will be integrated into the processes of the organization for which it is being created.

- The design of the information system as a whole is the task **of the information system architects**. They answer the question **WHAT**: what should the system look like, what functions should it have, and how should it be structured to fulfill its purpose. They design the structure, components, interfaces, and technologies with regard to functionality, security, performance, and future development. Last but not least, they ensure the technical feasibility of the design (similar to how an architect in construction takes statics into account).
- **Information system implementers (programmers)** are responsible for the question of **HOW** to technically implement (program) the architectural design. They ensure the correct and robust assembly of individual parts of the system (code writing, testing, deployment) so that the result is stable, secure, and corresponds to the architect's intention.

Architects and implementers use clear communication tools for effective collaboration. These include various standardized diagrams and visualizations that define the structure and relationships in the system at various levels of detail. Over time, a number of standards (e.g., UML, SysML, etc.), architectural principles, and proven design patterns for writing code have emerged to maintain consistency and reduce the mental effort (i.e., cognitive load) of the programming team.

1.2 Pitfalls of cooperation between architects and implementers of information systems

Unlike in construction (where even small houses are usually built according to a specific, albeit standardized, design), collaboration between architects and implementers (programmers) in the construction of information systems (IS) presents a number of challenges:

- **Poor or lack of cooperation with the architect:** The result is the creation of systems that are "glued together like a swallow's nest" and can hardly be modified according to new user requirements.
- **Incomplete architecture design:** Underestimating cooperation with future users leads to the system not fulfilling all of the customer's expectations. New requirements then emerge during pilot testing or even during actual use.
- **Overlapping design and implementation phases:** In practice, these phases often overlap, leading to constant changes during construction. Programmers make **ad hoc changes to the architecture** without consulting the architect, usually under pressure to deliver functionality

quickly. This makes the system inconsistent and creates **technical debt**, which reduces its stability and maintainability.

- **Underestimating requirements analysis:** When constructing buildings, the architectural design is selected first, and only then the contractor. In IT, analyzing **what** the user needs (**requirements analysis**) is a key part of the design. However, this phase is often overlooked, and a single tender is issued for both design and implementation, often with vague specifications directly from the user. This creates space for so-called "system integrators" who provide both services (both architecture design and its subsequent implementation). While this consolidation is effective, it carries a risk: integrators tend to propose solutions that favor their own tools and technologies, which may not be optimal for the customer. Figuratively speaking, it is similar to a construction company winning a tender for a church and—because it owns a panel factory—building a sanctuary out of concrete panels.
- **Vendor lock-in:** Vendors, especially those who provide both architecture and software solutions, tend to create customer dependence on the exclusive use of their products and services. This means that the customer is "**locked in**" to the original vendor for any modification or further development of information services. Switching to another supplier is then very difficult (or even impossible) without the risk of significant **financial** losses, technical complications, or operational disruptions. In extreme cases, it may be worthwhile to start building the entire information system from scratch. A clear example of this phenomenon is the fate of the first version of the eReceipt information system, which we will discuss later (see Chapter 2.1).

When building large-scale IS, which undoubtedly includes healthcare systems, it is necessary to avoid these pitfalls. Close cooperation between architects and future users is particularly important.

The design is not just about creating a plan for programmers. The key is to first clearly analyze **WHAT is the goal of informatization of the entire system** and its individual participants – stakeholders. It is necessary to analyze the processes within and between these parties, i.e., to perform a so-called **process analysis**. The result is the creation of a so-called "business layer" of a multi-layered architecture (Fig. 1). The word "business" here is not related to commerce; it is a terminological convention that in this context refers more to processes and operations. It would be more understandable in Czech to speak of a **process (or operational) layer** of architecture.

It is precisely the process analysis and design of the operational layer that require **the closest cooperation** between **software architects** and **future users**. However, this cooperation is particularly challenging in healthcare.

1.3 Cooperation between IT specialists and healthcare professionals in building healthcare information systems

Processes in healthcare are complex. Their creation places great demands on multidisciplinary understanding between healthcare professionals and IT specialists.

Healthcare and information technology are two different worlds, which leads to differences in the thinking and professional focus of their workers. Both sectors are constantly specializing. Leading analysts or system architects often lack in-depth

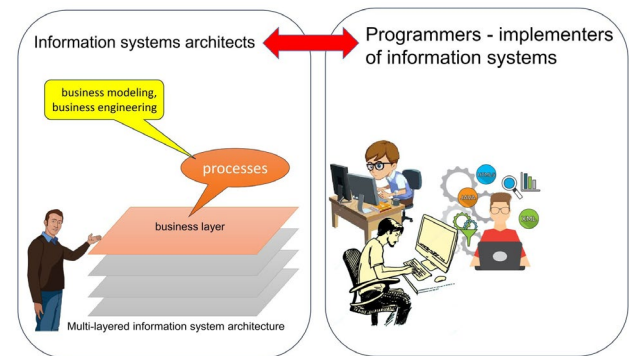


Figure 1 – Cooperation between information system architects and programmers implementing the architectural design. The role of architects is primarily to clearly analyze **WHAT** the goal of the information system is, what the requirements of its future users are, and how the IS will be integrated into the operations and processes of the entire organization. Based on this analysis, they propose the optimal structure for these processes. It is crucial to correctly design the so-called operational (process) layer of the entire multi-layer architecture so that the created system fulfills the purpose and needs of its users. Architects define components, interfaces, and their interrelationships, creating a clear and concise plan according to which programmers then implement the software architecture design. The task of programmers is then to find a way to optimally implement the designed architecture from a technical standpoint.

knowledge of healthcare processes, even in areas closely related to IT, such as electronic prescribing, the use of image analysis and machine learning for diagnostics, and other areas of eHealth. Similarly, doctors and healthcare professionals are closer to the natural sciences than to the world of algorithms, databases, and coding.

For the effective creation of information systems, it is therefore essential to moderate discussions between all stakeholders in healthcare (doctors, pharmacists, insurance companies, patient associations, etc.). The resulting consensus and agreements must be transformed into a precise and formalized form that will serve as:

1. **An architectural basis** for the implementation of information systems.
2. **A comprehensible description** for all participating experts.
3. The primary basis for the creation of **follow-up legislation**.
4. This approach is essential to ensure consistency between practical needs, technological architecture, and the existing legal framework.

In addition, healthcare information systems are closely linked to state and public administration systems, which brings further problems associated with the need for legislative safeguards.

1.4 eHealth and the legislative loop: how to protect sensitive data (and why it must be in the law)

The development of **state information systems** in democratic countries is subject to fundamental legal restrictions. While citizens can do anything that is not prohibited by law, **the state may only do what is expressly permitted by law**.

This means that every activity of information systems used by public authorities must be enshrined in law. **The law must describe precisely** where, when, and how the information

system works with sensitive data. The aim is to prevent data leaks and ensure that the state does not become "Big Brother" through its systems.

This applies in particular to healthcare information systems. The digitization of healthcare enables data sharing between healthcare providers, access to digital records, and the creation of clinical and administrative registries for healthcare management based on big data. All these processes must be ensured at the state level, where public authorities can only act within the limits of the law. Since healthcare involves working with highly sensitive data, it is necessary to protect patient privacy when storing and sharing such data and to eliminate the risk of unauthorized entities gaining access to it.

The digitization of healthcare thus places considerable demands on the creation of **high-quality legislation**, especially given the complexity of the processes connecting experts from different fields. Current trends require not only technological innovation, but also consistent communication across professions to ensure the effective implementation of new systems in practice.

Legislation for eHealth has two basic tasks:

- 1. Process description:** It must precisely and thoroughly define how the systems will be used, as agreed upon by architects and healthcare professionals.
- 2. Data protection:** It must strictly ensure the protection of personal data, both organizationally and through software architecture, and enshrine this security in legislation.

When creating regulations, **three groups of participants** communicate with each other, whose education and approach may seem incompatible at first glance: **healthcare professionals** (users of digitization tools), **IT specialists** (programmers and architects), and **legislators** (creators of the regulatory framework) – see Fig. 2.



Figure 2 – When creating healthcare information systems, the key issue is interdisciplinary understanding between healthcare professionals, IT specialists, and legislators.

For their communication to be effective, it is necessary to use **a common tool** that serves to **visualize** processes and objects from the outset. This tool must be **understandable** to all parties and, at the same time, it must enable the detection of errors or blind spots that are difficult to detect in complex legislative text through simulation before the software solution is created. **Process modeling** has proven to be the ideal solution for visualizing, simulating, and optimizing the steps contained in legislation.

2 Our experience with the creation of electronic prescription and electronic vaccination card systems

2.1 The emergence of electronic prescribing in the Czech Republic

The first attempts at electronic prescribing of medicinal products in the Czech Republic date back to 2006. In 2007, the State Institute for Drug Control (SÚKL) was entrusted with the development and administration of the eRecept system on the basis of Act No. 378/2007 Coll. on medicinal products. The possibility of prescribing medicines electronically was introduced in 2011, but initially, doctors and pharmacists did not use the system very much. Due to low interest, a political decision was made to **make eRecept mandatory** for all doctors and pharmacies from **January 2015**.

The supplier of software (including the eRecept system) for SÚKL was Tronevia (and its predecessors), which proved problematic in 2015. After the service contracts expired and at a time when SÚKL was preparing to launch new tenders, Tronevia began to assert copyright claims to the software applications. It demanded large sums of money for the copyrighted work and its use, originally up to CZK 167 million. Based on Tronevia's lawsuit, the Municipal Court in Prague issued a preliminary injunction that prevented SÚKL from accessing certain applications, including eRecept (specifically, the then-current version of the Central Repository of Electronic Prescriptions). The Minister of Health (Svatopluk Němeček) and the management of SÚKL sharply rejected the company's demands at the time as unreasonable financial "blackmail." SÚKL insisted that it had paid properly for the services provided, and the **deadline for mandatory use of eRecept** was postponed to **January 2018**.

Due to the court block, SÚKL had to quickly develop and deploy a new, replacement electronic prescription system to ensure functionality from March 2016. This definitively cut it off from Tronevia's software. The dispute with the company was finally settled by court settlement in 2022. Based on the settlement, SÚKL paid Tronevia compensation of almost CZK 14.5 million without acknowledging the original demand for an apology or the full amount of copyright claims. SÚKL had long since stopped using the systems involved in the dispute at that time.

2.2 Expansion of eRecept functionality

In **December 2017**, when the author of this article, Adam Vojtěch, was appointed Minister of Health, the number of doctors using electronic prescriptions remained low, in the order of a few percent. In addition, from **January 2018**, doctors faced financial penalties for not using eRecept, which caused considerable tension in the healthcare community. The main reason for the resistance was that neither doctors nor pharmacists saw any significant advantage in electronic prescriptions over traditional ones. In addition, most doctors at that time did not have a qualified electronic signature, which was necessary for identity verification.

In order for electronic prescriptions to make real sense from a healthcare perspective (enabling the assessment of drug interactions or the selection of the most appropriate pharmacotherapy) and not just be a mere replacement of paper with an electronic code, it was necessary for both doctors and pharmacists to have an overview **of all medications prescribed and dispensed to a given patient**. It was therefore necessary to expand the eRecept system with additional functionalities. This required both changes in the software architecture and the creation of related changes in legislation (Fig. 3).

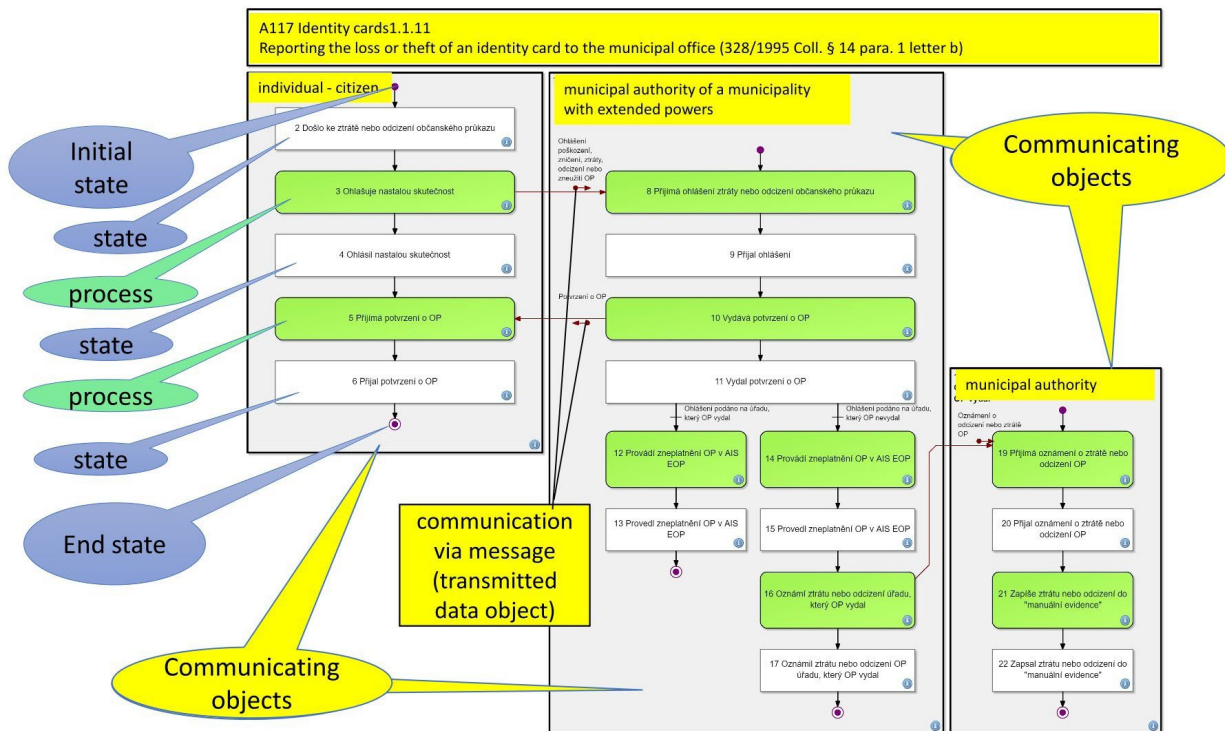


Figure 3 – In January 2018, according to the legislation at the time, it was necessary to start using e-prescriptions. However, most doctors at that time did not have the qualified electronic signature required for electronic prescribing. Penalties for not using e-prescriptions were postponed for a year, and the system had to be expanded with additional functionalities (especially patient medication records) that gave electronic prescribing real medical meaning.

During the preparation of the new functionality, it was necessary **to thoroughly discuss** the electronic prescription processes with all interested parties – the Czech Medical Chamber, representatives of pharmacists, insurance companies, the professional public, software architecture developers, and administrators of basic eGovernment registers.

Based on the results of these discussions, it was possible **to modify the existing software** and at the same time draft **new legislation** that described in detail all the processes related to the expansion of e-prescriptions (the so-called medication record). Subsequently, a draft amendment to the law was prepared, which passed through the approval process in the Czech Parliament.

Under normal circumstances, this process takes approximately two years. We decided to speed it up by using visualization technology to represent the structure of interconnected objects and processes. This method greatly facilitated interdisciplinary understanding between programmers, healthcare professionals, and legislators.

2.3 Visualization of the design of the new software and legislative structure in the BORM methodology

We chose the **Business Object Relationship Modelling (BORM)** methodology [1–7], which appeared in the early 1990s, as the basis for visualisation. It was developed as part of the Czech-British VAPPIENS project funded by the British Council, which provided it with a solid theoretical foundation and early conceptualisation. A key phase was the connection with the consulting firm Deloitte in 1996, which began to actively support the methodology in cooperation with the academic sector and use it in large-scale projects.

The main strength of BORM is its use of **very simple graphical notation** that is easy to understand even for non-technical users. This principle solves a common problem in system development: the difficulty that stakeholders (especially future users) face when designing **process architecture**, as they may

not always understand the complex technical aspects. The goal is to enable them to understand how the system will work and, based on discussion, arrive at its optimal design.

The practical application and dissemination of the BORM methodology are closely linked to the **Craft.CASE** software tool [8], which was specially developed to support this methodology. **Jiří Berger** is behind its development and patented technology [9]. Craft.CASE is, figuratively speaking, a kind of imaginary “pencil” for designing and modeling the process architecture of a system.

Craft.CASE is not only a tool for displaying architecture, but also serves as a modeling environment that allows you to test the behavior of the proposed system and detect errors in the design in a timely manner. This is of particular importance in non-technical systems, where the biggest challenge is interdisciplinary understanding between architects and users. The advantage of a clear visualization of processes is combined here with the ability to test their behavior using a simulation model.

As an example, we can mention the use of Craft.CASE in the design of civil code legislation (see Fig. 4). The process model is displayed here using communicating objects, which are characterized by a certain state at a given time. The transition of an object from one state to another is the result of a relevant activity (process) and may be conditional on the fulfillment of specified conditions. The start of a process can also be initiated by an activity in another object. In this way, objects pass data between each other and communicate. This made it possible to formally describe the processes addressed by the proposed legislation in an understandable way.

Essentially, the Craft.CASE tool made it possible to use simulation to test the behavior of all elements—for example, to determine whether a process would find itself waiting in vain for information from another object. This revealed a deadlock, where the sequence of transitions in the object stops. In the design of the Civil Code, process modeling made it possible to detect a number of legislative errors during its preparation.

Our experience



Figure 4 – The Craft.CASE software tool implementing the BORM method was tested in practice during the creation of the Civil Code. The process model is visualized using communicating objects that are characterized by a certain state at a given time. The transition from one state to another is the result of the action of the relevant process and may be conditional on the fulfillment of specified conditions. The process can also be initiated by an activity—a process running in another object—in this way, objects communicate with each other and exchange data. States in objects should gradually transition from the initial state to the final state. Craft.CASE allows this communication and state transitions to be simulated. If the design is flawed, the object may get stuck waiting in vain for information from another object; thanks to simulation, this error is detected before the text of the law is finalized. In this way, it was possible to identify a number of legislative shortcomings already during the preparation of the draft.

2.4 Use of the Craft.CASE tool and BORM technology to design software and legislative architecture to improve the functionality of eRecept

Craft.CASE development tool:

- is an interactive tool for the gradual creation of descriptions of objects, processes, and their interconnections;
- enables accurate description of process architecture;
- serves as a visualization tool that facilitates interdisciplinary understanding;
- enables the simulation of the behavior of interconnected objects and processes;
- creates accurate documentation for the preparation of legislation and for the design of the process architecture of software solutions.

That is why Craft.CASE was chosen as the basic development tool for designing the new software and legislative version of eRecept. Its author, Jiří Berger, was a key member of the development team.

We organized a series of intensive workshops to create a new eRecept architecture with extended medication record functionality. All interested parties participated in these workshops, from software developers to representatives of pharmacies, insurance companies, and legislators, as well as the Czech Medical Chamber and other experts. Based on these discussions, we gradually modified the visualized process architecture design in the Craft.CASE system (see Fig. 5).

The summary graphic diagram generated by the Craft.CASE system accurately describes the processes of the participants and their mutual influence. Detailed scenarios are stored in

the background of these diagrams, which made it possible to use simulation to verify that the proposed structure does not contain logical contradictions and behaves according to the original intentions.



Figure 5 – At the beginning of 2018, we organized a series of intensive workshops attended by all stakeholders (from software developers to representatives of pharmacies, insurance companies, and legislators to representatives of the medical chamber and other experts). Together, we discussed and modified the visualized design of the e-prescription process architecture, created using the Craft.CASE tool according to the BORM methodology.

During the workshops, changes were gradually introduced into the architecture, which were reflected in the next version of the proposal. The proposal thus evolved from its original form (Fig. 6) to the final, tenth version (Fig. 7), which was approved by all parties. This version became the springboard for modifying the software and creating legislation describing the chosen architecture. The diagram was also part of the explanatory memorandum to the draft amendment to the Medicines Act.

The solution also included a set of process diagrams describing individual scenarios in detail. For example, Figure 8 shows the process of prescribing a medicinal product, including the states and processes in communicating objects. This diagram is not just a passive image, but interactively simulates relationships, thereby testing the correctness of the entire model.

Thanks to this approach, it was possible to prepare the legislation and modify the software for the introduction of patient medication records in record time [10–13]. In December 2019, the law was passed by the Czech Parliament and the system became fully operational.

The introduction of e-prescriptions in 2018 was a turning point. The graph in Figure 9 shows how the combination of legal obligations and new benefits catapulted the use of the system. While electronic prescriptions accounted for a few percent at the end of 2017, their share rose to 80% in 2018 and exceeded 95% in 2019.

At the beginning of 2020, the COVID-19 pandemic broke out – thanks to the functional eRecept, chronic patients were able to access their medications remotely, which greatly helped the Czech healthcare system during a critical time.

2.5 Creation of legislation for electronic vaccination cards

A key part of electronic prescribing is the medication record, which contains an overview of all medications prescribed and dispensed to the patient. This allows doctors and pharmacists to assess drug interactions and incompatibilities. In addition, this system allows for integration with other applications, such as personalized prescribing.

While electronic records of COVID-19 vaccinations were hastily introduced with the onset of the pandemic, other vaccinations were not digitized. Records of these remained scattered in paper documentation at individual doctors' offices, and patients did not have electronic access to them.

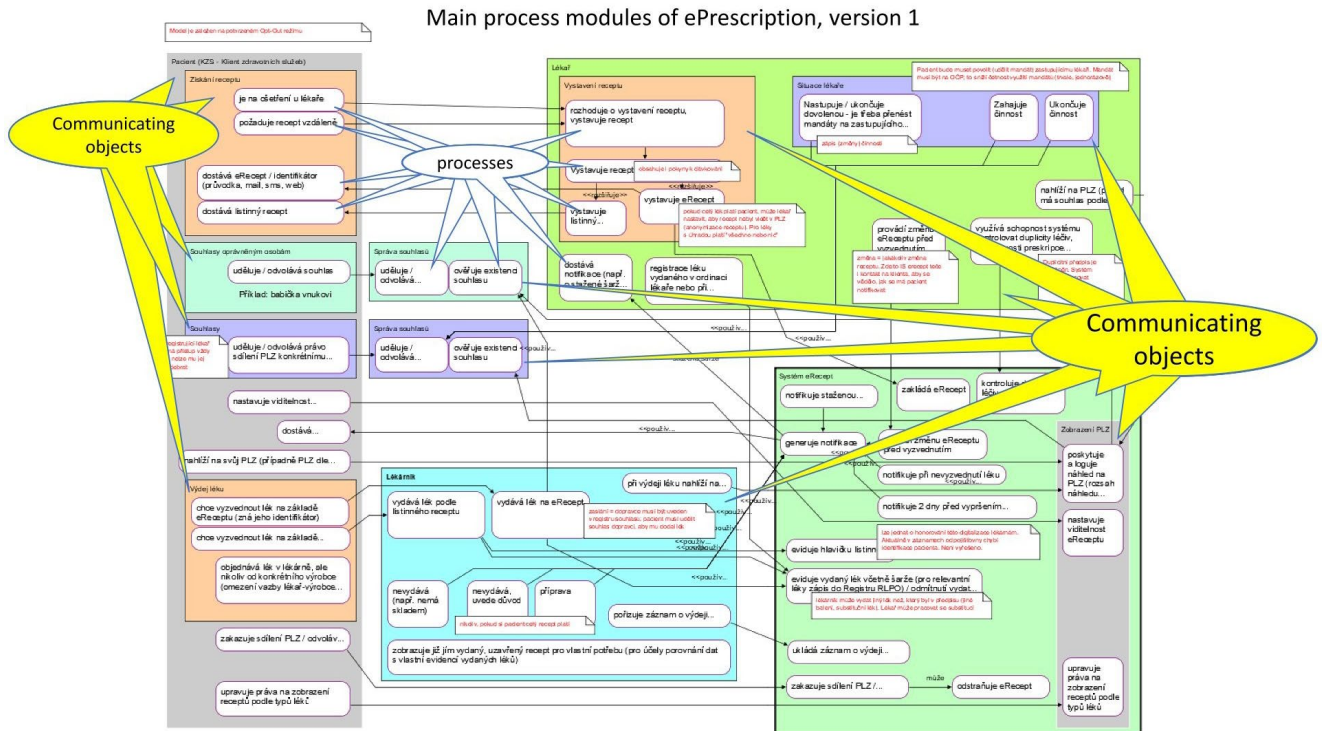


Figure 6 – First version of the eRecept process architecture.

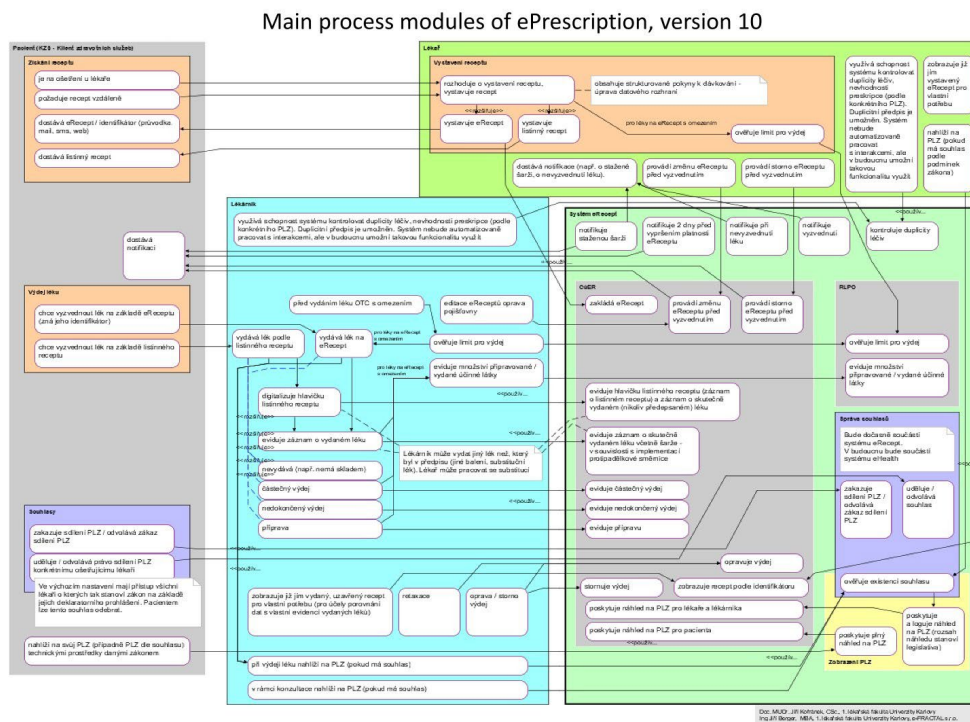


Figure 7 – Tenth, final version of the eRecept process architecture.

We therefore decided to expand the eRecept system and the associated medication record to include records of all vaccinations received and to create the necessary legislative conditions for its implementation. The electronic vaccination card can work on the same principle as the medication record. When designing the legislation and process architecture, it was possible to build on the proven structure of the electronic prescription, which only needed to be supplemented with specifics related to vaccination. As a result, the process model for the vaccination record was developed in just three weeks.

Figure 10 shows the structure of the proposed e-prescription process architecture, extended to include legislative support for

the electronic vaccination card. Based on this model, a legislative proposal was subsequently drafted, which MP Adam Vojtěch (who was no longer Minister of Health at the time) submitted on April 13, 2021, as a parliamentary amendment to the amendment to the Act on Addictive Substances. On June 2, 2021, the Chamber of Deputies approved it in its third reading.

The actual preparation of the bill, from the creation of the process model to the final legislative text, took about four weeks, and the bill was approved in just three months. This shows how beneficial process modeling and the associated formalization of process descriptions can be for the effective preparation of legislation [14].

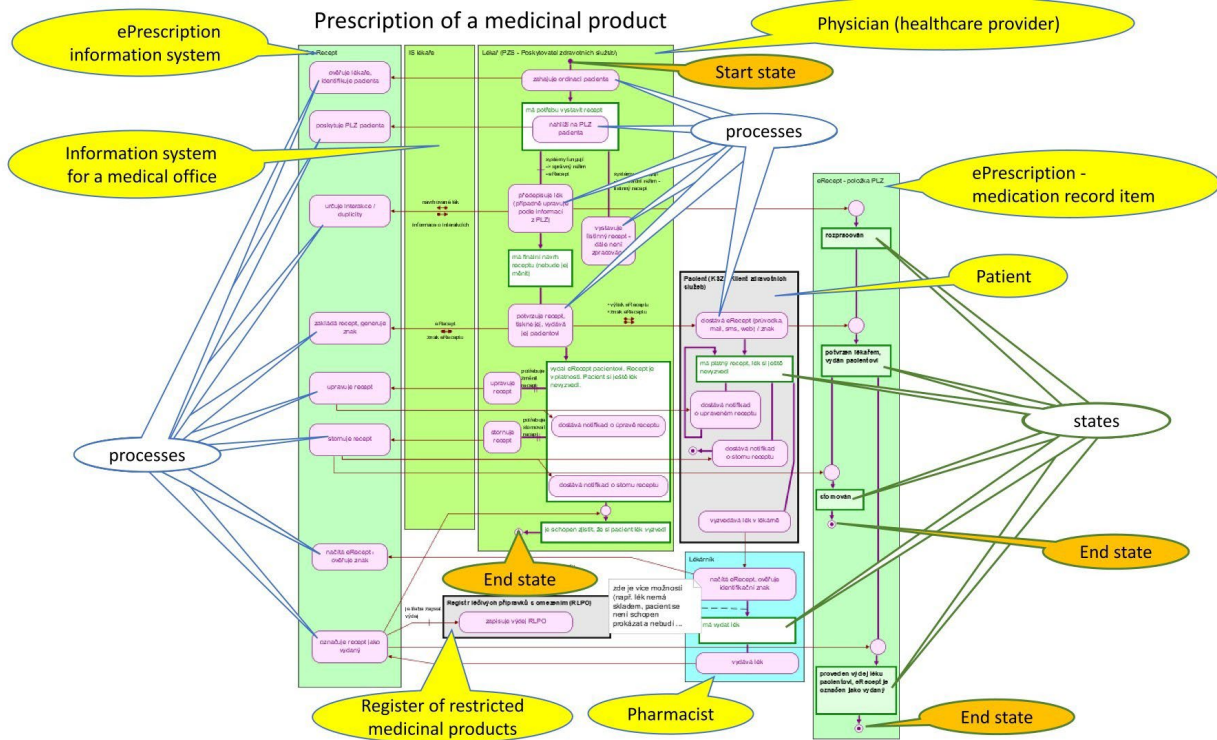


Figure 8 – Process diagram describing the prescription of a medicinal product, individual states, and processes in communicating objects. Simulation in the Craft.CASE tool made it possible to test and verify the dynamic behavior of this model.

Main process modules of ePrescription extended by electronic vaccination record

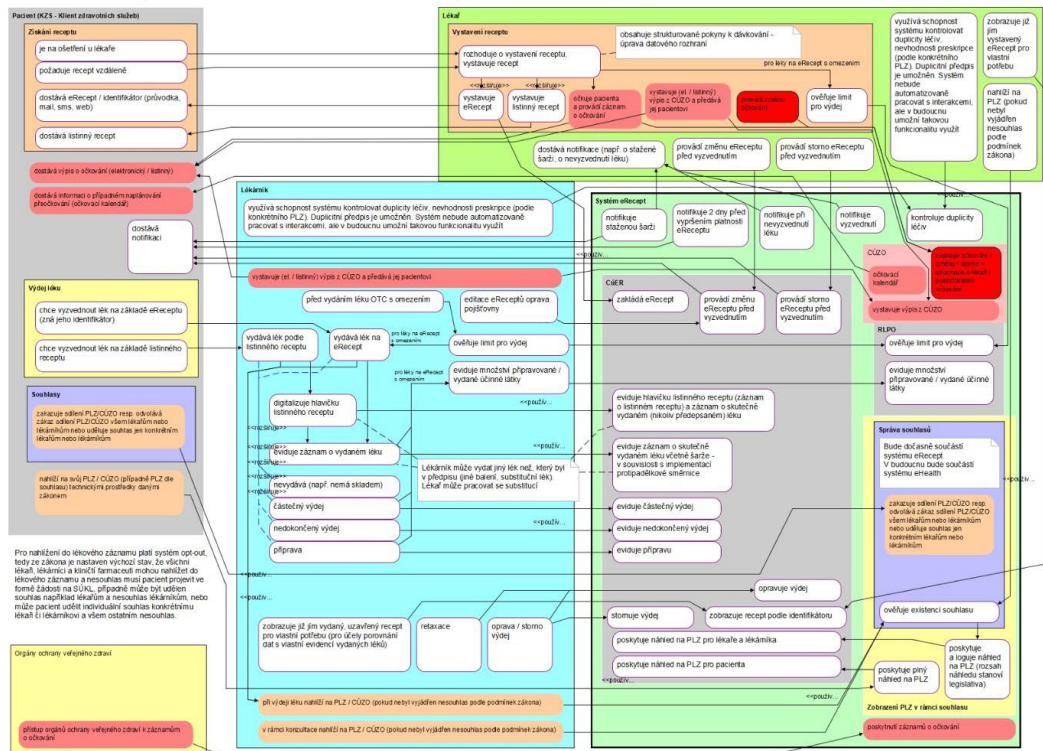


Figure 10 – Structure of the e-prescription process architecture design, extended to include legislative support for electronic vaccination records. In principle, this represents an extension of the patient’s medication record to include data on all vaccinations. Added objects and processes are highlighted in red.

The story of the Czech ePrescription

The introduction of mandatory ePrescription in 2018 was a turning point. The graph shows how the obligation catapulted the use of the system and the involvement of doctors and pharmacies.

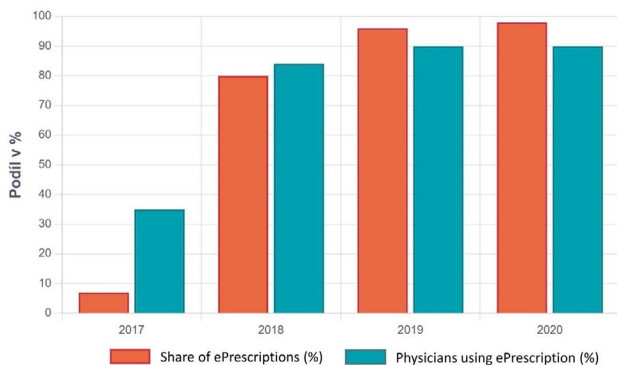


Figure 9 – At the end of 2017, the share of e-prescriptions was still in the single digits. A major turning point came in 2018 with the introduction of mandatory requirements and new functionalities; a year later, the share of electronic prescriptions exceeded 95%.

3 Formalization of legislation

3.1 Goals and paradigm of digital formalization of legislation

In short, formalization is a process in which we replace vague and ambiguous words or descriptions (such as those found in everyday speech) with precise symbols and formulas according to a defined system of rules. In this context, formalization means transcribing sentences from everyday English into symbolic notation, which can then be handled according to given rules. The goal is to eliminate ambiguities: while in everyday speech a single word can have multiple meanings, in a formal system each symbol has only one meaning. Formalization is particularly important in software development, where program behavior requirements are formalized into code according to the rules of the programming language.

The digital formalization of legislation, often referred to in a broader context as **computational law**, shifts the traditional concept from the theoretical to the technical and application level. It is no longer a matter of mere digitization, i.e., the simple conversion of the text of a law into electronic format (e.g., PDF), but of a profound and complex transformation of legal texts, rules, and processes into structured, machine-readable, and computationally processable formats [15].

This transformation uses formal models of law that allow not only the text to be represented, but also its internal logical structure, hierarchy, and explicit relationships between individual provisions. While 20th-century legal theory largely abandoned rigid formalism (recognizing that law is full of ambiguities, principles, and discretions that cannot be reduced to pure logic [16]), the advent of computer science enforces formal models. Digital formalization thus represents a pragmatic and engineering "reunion" with formalism. The goal is not to naively claim that all law is an algorithm, but rather to identify and accurately model those parts of it that are algorithmizable—in the Czech context referred to as "fully programmable cases" [17].

The primary goal of digital formalization is not only to make law accessible, but to integrate it into the digital systems of the state and society. This leads to a fundamental paradigm shift: from law as a passive source of information to law as an active system component. Traditional legal information systems were designed for legal search – their goal was to help users find the

relevant section [18]. Modern formalized systems are geared towards legal execution – their goal is to run the logic contained in a paragraph as an algorithm (e.g., "if income X and status Y, then entitlement Z"), as suggested by concepts such as **Rules as Code** [19].

The key objectives of this new paradigm are:

- **Interoperability:** The ability of different national and international systems (registries, parliaments, courts) to seamlessly share and correctly interpret legislative data. This is a basic prerequisite for the functioning of the EU's single digital market [20].
- **Automation:** The ability to automate a wide range of tasks, from routine consistency checks of draft legislation [21] to automated compliance checking [22] to direct digital enforcement of certain standards (e.g., tax calculation, environmental zone management) [23].
- **Decision Support:** Providing advanced tools for legislators to model the impact of proposed changes, while also providing more understandable tools for those affected by the standards [21].
- **Transparency:** Making the entire legislative process more transparent, from the initial draft to the final publication, and making it available to the public in a machine-readable form, e.g., through the eSbirka or eLegislativa portals.

3.2 Most commonly used standards and semantics of digital legislation

The **digital formalization** of legislation is not a uniform process; it is rather a "layered architecture" consisting of several technological levels. These layers are logically linked – from the basic structure of the document through its semantic meaning to its executable logic.

1. Structural formalization: XML and the Akoma Ntoso (AKN) standard

The basic building block is the standardization of document structure. Without a uniform format for describing what constitutes a "law," "section," or "reference," interoperability is impossible. The key global standard in this area is **Akoma Ntoso (AKN)** [20]. This is a standard of the international organization OASIS, which is part of the broader LegalDocumentML (LegalDocML) initiative. AKN is based on **XML (eXtensible Markup Language)** [24] and provides a detailed dictionary (schema) for semantic and structural tagging of legal, parliamentary, and court documents.

The purpose of AKN is not only technical, but primarily strategic. As stated in the specification, its goal is to define a common format and data model for the exchange of documents between institutions anywhere in the world [25]. It covers the entire document lifecycle – from parliamentary records to draft laws and approved legislation to court decisions. It thus enables the development of modern, component-based systems instead of outdated monoliths. In the EU context, AKN (in its specific variant "AKN4EU") is the basis for key tools such as the LEOS editor [26].

2. Semantic formalization: Legal ontology (RDF and OWL)

However, the XML structure (Akoma Ntoso) alone is not enough. AKN can describe that a certain text is a paragraph (<paragraph>) or a reference (<ref>), but it cannot define the meaning of the legal relationships and

concepts contained therein. This is where the second, semantic layer, based on semantic web technologies [27], comes in:

- **RDF (Resource Description Framework)** [28]: This is a basic data model. Instead of describing documents, it describes relationships using simple triples (subject–predicate–object). This allows the creation of linked knowledge graphs.
- **OWL (Web Ontology Language)** [29]: This is a language for defining ontologies, i.e., formal, machine-readable models of knowledge in a specific domain. OWL (built on top of RDF) allows you to define classes (e.g., "Contract," "Obligation"), their properties, and their logical relationships (e.g., "A purchase contract is a type of contract").

In a legal context, ontologies enable the modeling of complex legal concepts, hierarchies of norms, and inference rules. This is a necessary prerequisite for advanced legal analysis systems, intelligent search, and artificial intelligence applications that need to "understand" the meaning of text, not just its structure [30].

Work on legal ontologies has a tradition in the Czech Republic, as evidenced, for example, by the "Legal Electronic Dictionary (PES)" project [31]. This is a long-term interdisciplinary project (collaboration between lawyers, linguists, mathematicians, and computer scientists) focused on the analysis of legal terminology. From the point of view of formalization, the approach chosen by the project is key: instead of attempting to create a single universal legal ontology a priori (top-down), which would inevitably be subjective and difficult to apply, the creators have taken the path of building ontologies "from the bottom up" (bottom-up). This means that the structure of terms and their relationships is derived directly from the analysis of specific texts (e.g., law textbooks). This approach better respects the contextual nature of legal language.

The most commonly used technologies are shown in Table 1.

Attribute	XML / Akoma Ntoso	RDF / OWL	Rules as Code (RaC)
Main purpose	Document structure and interoperability	Knowledge representation and inference	Automation and execution of rules
Basic unit	XML element (e.g., <article>)	Triple (Subject–Predicate–Object)	Logical statement (IF-THEN-ELSE)
Example in law	Tagged Term xt of the law (eCollection)	Legal ontology (relationships between norms)	Code for calculating benefit entitlement
Sources	[20]	[28,29]	[

Table 1 – Comparison of technologies used for digital formalization of legislation.

3.3 Process analysis and process modeling – a tool for interdisciplinary communication and a basis for legislation

Digital formalization does not only concern the final product (the law), but increasingly also the process of its creation. The creation of a legislative framework is not only a matter for legislators, but for all interest groups affected by the given

standard. When creating new regulations, it is therefore advisable to use communication mechanisms that facilitate mutual understanding.

This is particularly important for regulations concerning the digitization of public administration. Since public authorities can only do what is expressly permitted by law when digitizing administrative agendas (the principle of enumerative public law claims), legislation must describe the processes of informatization in a completely unambiguous manner.

This principle is absolutely crucial in the informatization of healthcare, where highly sensitive personal data is handled. When storing and sharing data, it is necessary to avoid the "Big Brother effect," i.e., to protect patient privacy and eliminate the risk of unauthorized entities gaining access to the data. In order to define these processes precisely in legislation, they must first be described in a form that can be understood by system users, IT specialists, and legislators. Formalization is not a means of simplifying the writing of standards, but primarily a tool for communication and interdisciplinary understanding.

For effective communication, tools for visualizing processes and objects must be used from the outset. This description must be understandable to all parties involved. Standardized graphical notations, originally derived from **UML** (Unified Modeling Language) [32], are used for this purpose. These include, in particular, **SysML** (Systems Modeling Language) [33], adapted for the description of complex systems involving hardware, software, data, and people. **BPMN** (Business Process Model and Notation) [7] is widely used for process analysis, its main advantage being its comprehensibility for analysts, developers, and end users.

A number of tools have been developed for working with SysML and BPMN, ranging from commercial systems such as **Enterprise Architect** [34] to open tools such as **Archi** (for the **ArchiMate language**) [35]. The results of process modeling of public administration agendas are publicly available in the Czech Republic on the website of the **Digital and Information Agency (DIA)** [36].

The Business Object Relationship Modelling (**BORM**) methodology [1–7] represents a structured approach to analysis with a special emphasis on interconnected processes and objects. It combines an object-oriented approach (states and relationships) with process modelling (time sequence and dynamic behaviour). The main strength of BORM is its very simple notation, which helps even non-technical parties understand the complex diagrams commonly used to describe architecture.

BORM is based on precise mathematical foundations (finite automata and Petri nets). Thanks to this robust formal background, models can not only be created, but also simulated, verified, and validated. The model works on the principle of communicating objects. A change of state is triggered by an activity (process), which may be conditional on the fulfillment of criteria or initiated by another object. Through mutual communication and data transfer, a clear description of complex systems is achieved.

The **Craft.CASE** software tool [8] allows these diagrams to be not only created, but also tested for behavior using simulation. This makes it possible to detect in a timely manner whether a deadlock is occurring, where a process waits in vain for a signal from another object and stops.

In solving the ePrescription issue, we have verified in practice that the BORM method and the Craft.CASE tool have greatly facilitated understanding between healthcare professionals, legislators, and developers. This allowed us to arrive at a final solution in several iterations, according to which legislation was drafted, software was updated, and user work methodology was updated [10–14].

4 Object-Process Methodology (OPM)

BORM technology and the Craft.CASE development tool enabled us to gradually create a process model based on process analysis, display it in the form of diagrams, and verify its behavior using simulation.

However, the opposite approach is also necessary: based on the current state—i.e., a textual description of the legislation and a standardized description of the software system architecture—create, for example, using the generative capabilities of large language models (LLM) of artificial intelligence (AI), a process model (Fig. 11) that could be further modified.

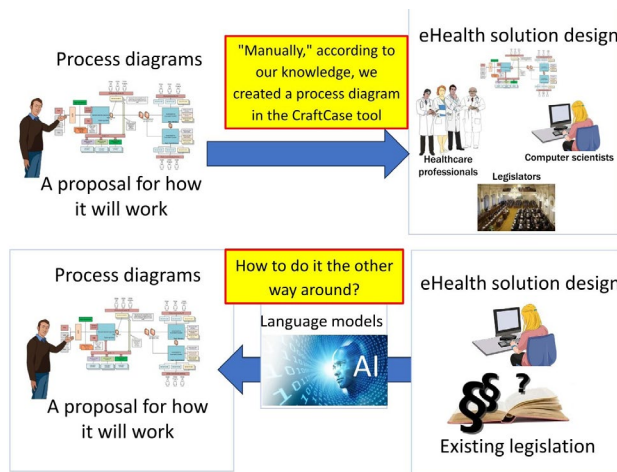


Figure 11 – Using the Craft.CASE tool, we were able to conveniently create process diagrams "manually" and verify their behavior through simulation. This raises the question of how to use artificial intelligence language models to do the opposite—that is, how to automatically generate a process model from existing legislation and architecture descriptions.

Here, it is possible to use **Object-Process Methodology (OPM)**, which serves as a comprehensive language and methodology for conceptual system modeling. OPM was developed by Professor Dov Dori, with the initial ideas published in 1995 in an article dealing with the conversion of engineering drawings into 3D CAD models [37]. Early applications of OPM demonstrated its versatility in various fields, including computer-integrated manufacturing, image analysis, e-commerce transactions, and research and development environments [38,39].

The first book on OPM, *Object-Process Methodology: A Holistic Systems Paradigm*, was published by Springer in 2002 [40]. Four years later, another textbook followed [41], in which Dov Dori presents the theoretical and practical application of this methodology in the design and modeling of complex systems using numerous examples.

These publications marked a significant step in the formalization and dissemination of OPM principles. The origin of this methodology in solving real engineering problems, rather than in purely theoretical abstraction, gives it a pragmatic and robust foundation. Its rapid application in various fields even before its formal standardization confirms its inherent usefulness and adaptability. The publication of both books provided a comprehensive and accessible foundation that was key to building a professional community and facilitated the wider adoption of the methodology, whose international ISO standard was completed in 2024.

4.1 Holistic principle, complexity management, and simulation capability

OPM offers a unified holistic (from the Greek holos – whole) view of the system as a whole – it integrates structure, functions, and behavior into a single coherent model. It can capture both the structure (objects) and behavior (processes) of the system in a single comprehensive model and integrates these two views into a single comprehensive object-process **diagram (OPD)**.

Using this diagram, we can clarify the behavior of the modeled system using simulation in a tool that implements OPM. Complexity is managed using hierarchical abstraction and refinement. Hierarchically linked object-process diagrams (OPDs) are used to capture the complexity of the entire system – from a single root diagram to detailed levels at lower hierarchical levels.

Instead of several specialized types of diagrams, as in other modeling languages (e.g., 9 to 14 types in UML and SysML), only one type of diagram is used here, which is hierarchically linked. Unlike other approaches, OPM provides a single, comprehensive view, which improves communication between stakeholders and provides a comprehensive picture of dependencies.

OPM enables automation and simulation and maintains a balanced view of objects and processes, distinguishing it from other modeling languages and expanding their capabilities.

4.2 Bimodal, graphical and textual representation

Essentially, when creating an **object-process diagram (OPD)**, a text description is automatically generated in a simple, normalized **object-process language (OPL)**. Both descriptions are interchangeable: an OPD diagram can be generated from a text description in OPL and vice versa. The modeled system is thus represented in two equivalent ways – visually and textually. This duality ensures both formal accuracy and intuitive understanding.

This offers the possibility of linking the OPL text description with large language models (LLM) in the future. Thanks to the bidirectional convertibility between OPL and OPD, it will be possible to link AI methods with exact simulations in a graphical model [42] (Fig. 12).

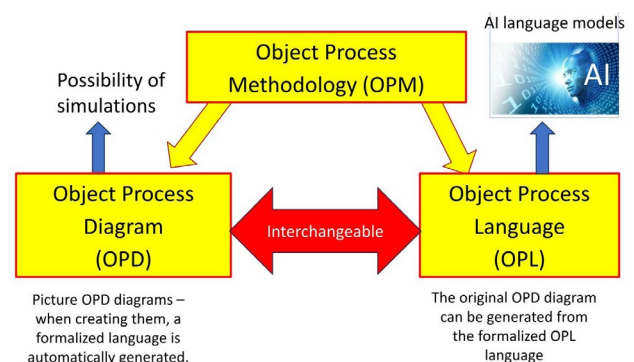


Figure 12 – Object-process methodology (OPM) defines a graphical representation of a complex system using hierarchically organized object-process diagrams (OPDs). A tool implementing this methodology (e.g., OPCLoud) allows the behavior of the modeled system to be simulated using OPD diagrams. Each object-process diagram has its text representation in the so-called object-process language (OPL). Both descriptions of the system (graphical OPD and text OPL) are interchangeable. This duality opens up the possibility of linking the text form of OPL with large AI language models in the future, while algorithmically simulating the behavior of the system using the graphical OPD model.

OPM combines aspects of 'language' and 'methodology'. As a language, it provides formal syntax and semantics for modeling, ensuring the accuracy and unambiguity of the system description. As a methodology, it offers a systematic way to apply this language in development and analysis. This combination addresses both the **WHAT** (what the system does) and **HOW** (how the system does it) questions, making OPM a comprehensive tool for practice. The ability to integrate different disciplines and represent all important interactions makes this methodology widely applicable in many different areas [43–46].

4.3 International standardization

The importance of OPM is underlined by its recent standardization in the form of the international standard **ISO 19450:2024** [47,48]. The Czech Agency for Standardization is already preparing a Czech version of this international standard, which is to be completed and published in February 2026, and we are personally involved in its Czech localization.

The standardization of OPM by the ISO standard increases its credibility and promotes wider acceptance, especially in legislatively regulated sectors where formal specifications are paramount. This brings the following benefits for system architecture design:

- **Improved understanding and communication:**

Graphical-textual (bimodal) representation makes complex models accessible to both technical and non-technical stakeholders, as it accommodates different ways of thinking (cognitive styles). The result is early stakeholder involvement, faster requirements gathering, and easier consensus on the design.

- **Reduced ambiguity and increased accuracy:** OPM's formal syntax, minimal ontology, and strict definitions minimize ambiguity, leading to clearer and more accurate system specifications.

- **Accurate analysis and design:** An integrated view of function, structure, and behavior in a single model provides a coherent reference framework, increasing the accuracy of both analysis and design.

- **Complexity management:** OPM's refinement and abstraction mechanisms allow modelers to manage large amounts of detail, prevent diagram clutter, and maintain clarity across different levels of abstraction.

The creation of the OPM international standard makes it possible to unify the tools for creating and using this technology, especially for the design and analysis of complex systems, including healthcare information systems [46].

4.4 Example – electronic prescription solved using OPM

We originally prepared the background materials for the draft legislation and software solution for eRecept using the BORM methodology in the Craft.CASE tool [10–14]. To illustrate the possibilities of using Object-Process Methodology (OPM) for architecture design, we will conclude with an example in the **OPCloud** environment. This is a software solution from the Israeli company OPcloud Ltd., designed for creating models according to the ISO 19450:2024 standard (www.opcloud.tech).

The structure of models created in OPM resembles Russian nesting dolls (Fig. 13). At the highest level, we work with one basic process, "Prescription of a medicinal product," to which six

Example: e-Prescription in OPD

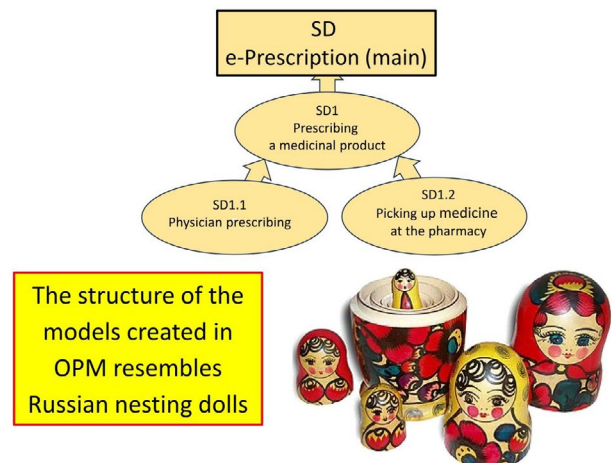


Figure 13 – Hierarchical structure of models created in OPM – when modeling eRecept, we move through three hierarchical levels.

objects are linked. In OPM, we work with entities (things), which can be either objects or processes, and with links between them.

Objects are entities that exist, arise, or cease to exist throughout the entire period under consideration; they are marked with a rectangle in the diagram. A distinction is made between whether they are tangible or intangible. Tangible objects (e.g., Patient, Doctor, Pharmacist, Medicine) have a shadow behind the rectangle. The eRecept information system or Patient Medication Record are intangible objects.

Processes are always temporary and transient and are marked with an ellipse. Like objects, they can take place in the tangible world (e.g., the process "Prescription of a medicinal product" has a shaded ellipse) or be intangible, taking place in the information world (e.g., searching for data in a database).

There are links between objects and processes that express their relationships. These are either structural or procedural. In this article, we will focus on **procedural links** that represent **the interaction between objects and processes**:

1. **Transformational links:** These express that objects can create, consume, or change the state of processes.

- a. The process "Prescription of a medicinal product" (Fig. 14) creates a new object "Medicine" using a **result link**, shown by an arrow from the process to the object.
- b. The "Doctor's activity" process creates the "ePrescription" object, which is consumed by the "Picking up medication at the pharmacy" process (Fig. 16) using a **consumption link**.
- c. A change in the state of an object (e.g., "Patient medication record") is shown by an **effect link** in the form of a bidirectional arrow.

2. **Enabling links:** These express that a process requires an object to activate it, but is not itself affected by it (e.g., "Doctor," "Patient," or "IS eRecept"). These links have a small circle on the object side:

- a. **Agent link:** If the enabling object is a living person ("Doctor," "Patient"), the circle is filled in.
- b. **Instrument link:** If it is an inanimate instrument ("IS eRecept"), the circle is empty (see Fig. 14).

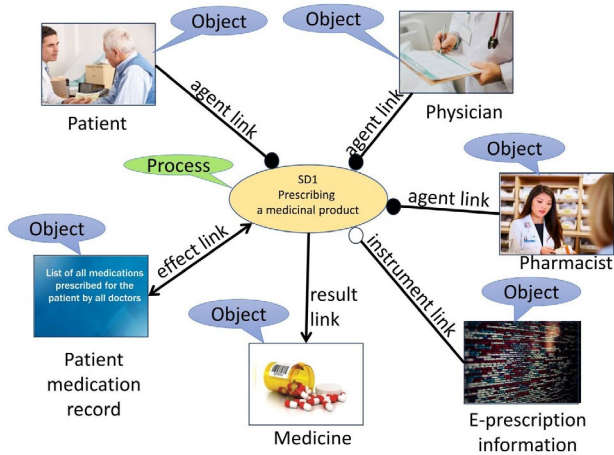


Figure 14 – Pictorial diagram of the highest level of the eReceipt model. The "Prescription of Medicinal Product" process is linked to five objects.

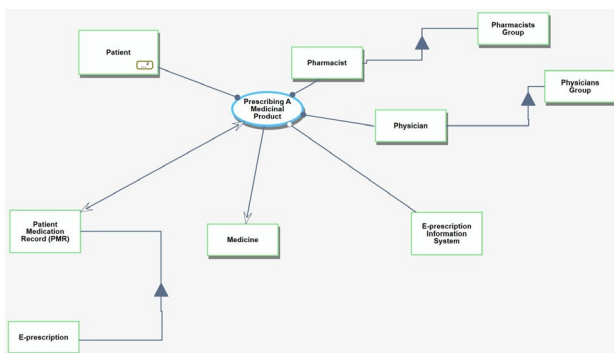


Figure 15 – Object process diagram (OPD) of the highest level created in the OPCLoud tool.

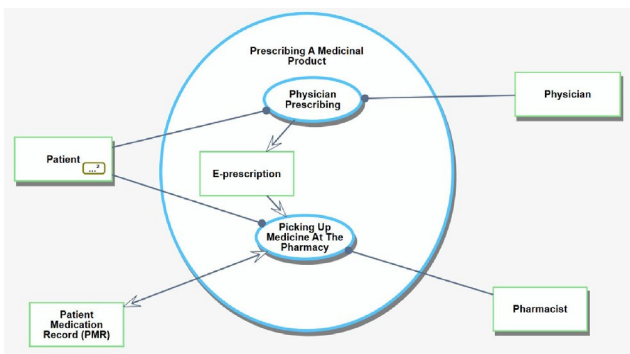


Figure 16 – The OPD of the "Prescription of Medicinal Product" process contains two processes: "Doctor's Activity" and "Pickup of Medicine at the Pharmacy." The first process creates the "E-prescription" object, which is consumed by the "Picking Up Medicine at the Pharmacy" process. This process influences the "Patient Medication Record" object through a transformation link.

Procedural links can also originate from individual object states. For example, the object "Patient" (Fig. 18) can have two states: either they pick up the medication themselves (thereby activating the "Medication Pickup" process), or they authorize another person to pick it up.

Figs. 15–18 show object-process diagrams illustrating a simplified model of the eReceipt architecture in OPM. For simplicity, the processes defining user settings for the states of some objects during modeling have been omitted.

The complexity of the model is controlled in OPM using hierarchical abstraction. The entire model is linked from the initial diagram (SD level) to the detailed levels (SD 1.1, SD 1.2). Each

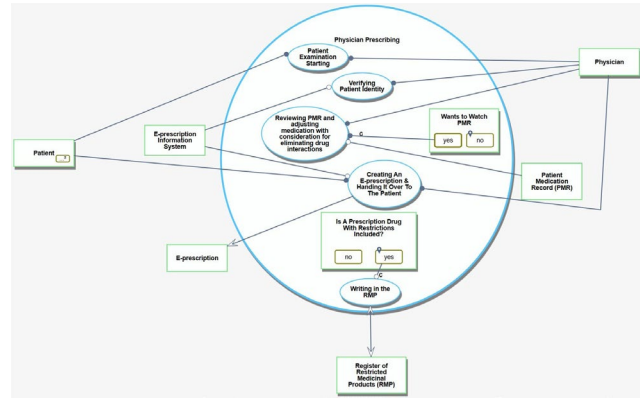


Figure 17 – OPD of the "Doctor's Activity" process.

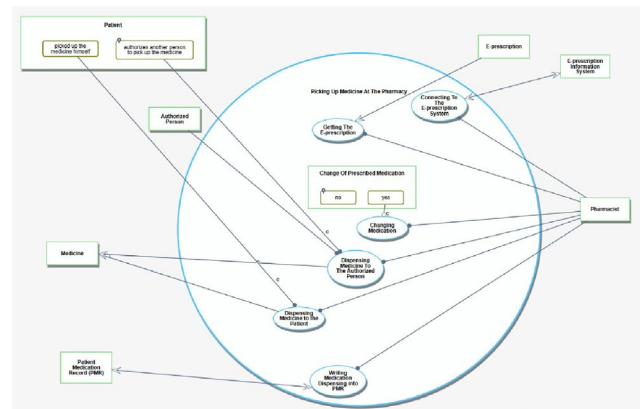


Figure 18 – OPD of the "Medicine Pickup at Pharmacy" process.

diagram (OPD) corresponds to a text representation in OPL (Object-Process Language), see example in Fig. 19.

The behavior of the model can be tested by simulation. Fig. 20 shows how simulation works in the OPCLoud tool. The actual simulation process is available on video: <https://www.youtube.com/watch?v=qWFBxIBFK6s>

- Model Name: e-Prescription in Czech Republic
SD
1. Patient is stateful.
2. Patient Medication Record consists of E-prescription.
3. Physicians Group consists of Physician.
4. Pharmacists Group consists of Pharmacist.
5. Patient, Pharmacist, and Physician handle Prescribing A Medicinal Product.
6. Prescribing A Medicinal Product requires E-prescription Information System.
7. Prescribing A Medicinal Product affects Patient Medication Record.
8. Prescribing A Medicinal Product yields Medicine.
- SD1: Prescribing A Medicinal Product in-zoomed
1. Prescribing A Medicinal Product from SD zooms in SD1 into Physician Prescribing, and Picking Up Medicine At The Pharmacy, which occur in that time sequence, as well as E-prescription.
2. Patient is stateful.
3. Patient and Physician handle Physician Prescribing.
4. Physician Prescribing yields E-prescription.
5. Patient and Pharmacist handle Picking Up Medicine At The Pharmacy.
6. Picking Up Medicine At The Pharmacy affects Patient Medication Record.
7. Picking Up Medicine At The Pharmacy consumes E-prescription.

Figure 19 – To illustrate the form of the OPL language, we present the linguistic form of the object process diagrams (OPDs) shown in Figures 15 and 16.

4.5 OPM as a validation metaframework against hallucinations of large language models (LLMs)

LLMs provide “human-sounding” textual descriptions of processes, but without being anchored in a human-controlled model, they can lack true understanding. For example, Klievtsova et al. [49] show that LLMs are capable of producing human-like descriptions of processes based on predefined patterns (e.g., process models in BPMN), but the descriptions they generate clearly lack a real understanding of process models, and a hybrid approach involving the work of a human analyst is necessary for their use.

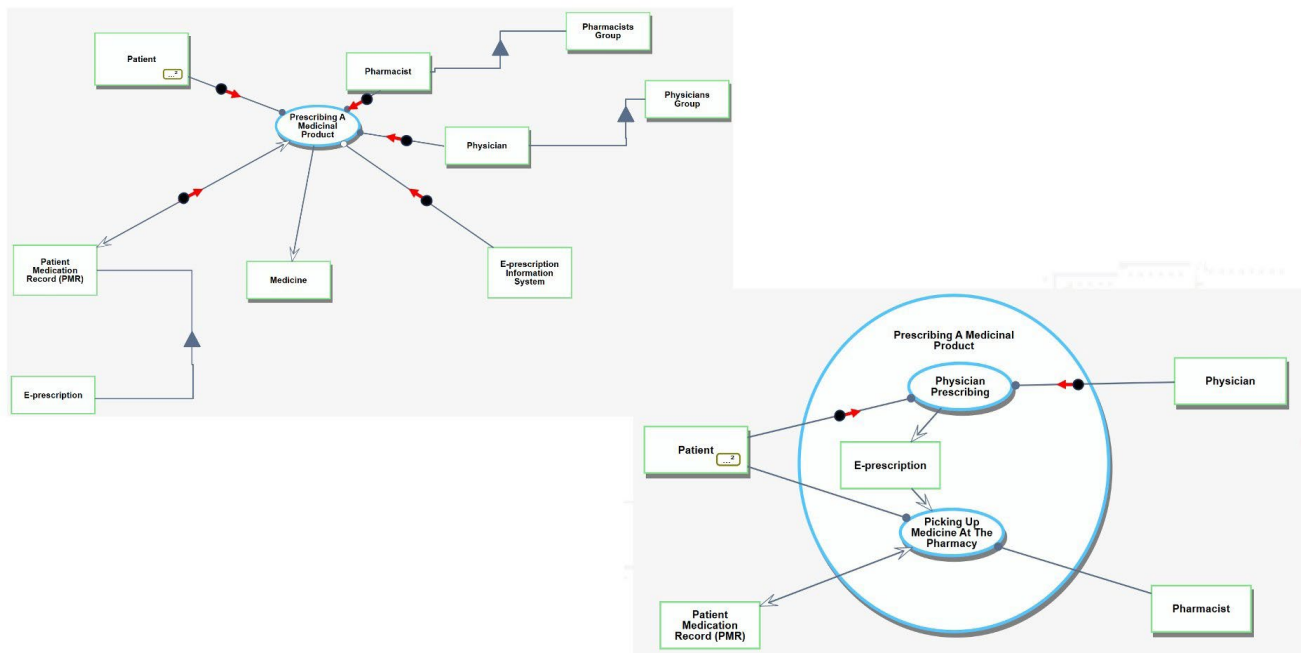


Figure 20 – Simulation of the behavior of the object-process model of eReceipt in the OPcloud environment. The animation illustrates the dynamics of the system: symbolic "balls" (tokens) move between objects and processes, gradually opening activated diagrams at lower levels of the hierarchy. This allows for detailed observation and validation of the model's behavior in real time.

OPM now offers two approaches that complement each other perfectly (see Fig. 12):

- **The enormous computing capacity of LLM** can be used as a language extractor, allowing us to analyze piles of text material and summarize knowledge into newly created text.
- **OPM as a normalizer, corrector, and validator**, summarizing text into a normalized form of standardized OPL language, which is linked to a graphical system of hierarchically organized (and potentially simulatable) OPD diagrams.

In the future, a well-configured LLM model using OPM should be able to quickly extract the necessary knowledge from text (workshop notes, methodology, descriptions of existing processes and legislation, and other text sources) and convert it into the precise syntax of OPL expressions such as:

- **objects** (e.g., Patient, ePrescription, Vaccination Record, Consent);
- **object states** (e.g., ePrescription: issued/cancelled/dispensed; Consent: granted/refused);
- **processes** (e.g., Prescribing a drug, Dispensing a drug, Vaccination, Vaccination record, Signing consent);
- and the relevant **structural and procedural links** between them.

The power of LLM models can be combined with validation in OPM using simulation in a modeling tool. The analyst can then immediately see the created OPL expressions in the OPD diagram, run the simulation, and verify and validate the model.

LLMs are prone to hallucinations because they can generate plausible-sounding statements without any guarantee of truthfulness. Such outputs from LLM that we consider hallucinations are:

1. **Logically unsolvable constructions** known as Russell's paradox [50]
2. **Syntactically correct but semantically empty sentences** known as Wittgensteinian "nonsense" [51]

LLMs have the enormous advantage of being able to propose quickly, but they cannot control themselves. However, OPM + human analysts add a filter that reduces the risk of hallucinations passing into the specification, which is theoretically consistent with Tarski's conclusion that semantic truth cannot be reliably enclosed within the same language system without added layers of metalanguage [52]

OPM, which is just such an added meta-layer to language models, allows a model generated from LLM in the graphical form of OPD to be run, simulated, and verified by a human analyst. This is extremely useful in the design of healthcare information systems, as it allows us to detect:

- **temporal nonsense** (e.g., "consent is given only after the procedure is performed"),
- **missing actors** (the process "happens by itself"),
- **non-existent objects** (LLM "invents" artifacts),
- or **hidden state conflicts** (the object is simultaneously in incompatible states).

LLMs are suitable for use as generators of candidate sentences in OPL, but not as an authority deciding on correctness. Practice shows that although LLMs can create credible human descriptions, they cannot fully understand the process structure and consistency of the model. The outputs must therefore always be compared with the OPM structure and verified by a human analyst. Simulation in the OPM model can then serve as a mandatory "anti-hallucination gate" before system implementation, as simulation acts as an effective error detector and validator.

This also corresponds to the experience of colleagues from the Technical University of Munich (TUM), who, as part of their benchmark, compared the outputs of the same LLM when working only with natural language input versus when supplementing the OPM model in the form of OPL to LLM. They showed that this integration, thanks to the introduced ontology and formal structure, leads to higher quality of generated outputs and a significant reduction in hallucinations of technical specifications and requirements for space mission subsystems [53].

5 Conclusion

Process modeling plays a key role in bridging communication gaps and differing perspectives between healthcare professionals, IT specialists, and legislators. The complexity of healthcare legislation and the different professional backgrounds of these groups often lead to barriers in understanding and inconsistent interpretations of legal norms. The use of tools for the formalized description of complex processes and their graphical visualization effectively removes these barriers. In addition, simulation allows for the behavior of the proposed architecture to be tested and errors in the design to be detected in a timely manner.

Using the example of creating e-prescriptions in Czech legislation, we have seen in the past that formalizing the design of software and legislative architecture using the BORM methodology and the Craft.CASE tool has significantly shortened the entire legislative process of preparing a law.

The new standardized **Object-Process Methodology (OPM)** localized into the Czech language and modern tools that implement this technology further expand these possibilities:

- **Creation of a common standardized language:** OPM provides a visual notation that is understandable to healthcare professionals, technical specialists, and legislators. This common language for describing processes and legal requirements prevents communication gaps and promotes shared understanding.
- **Visualization of complex interactions:** Modeling allows for the graphical representation of the behavior of the created process architecture, which helps stakeholders better understand operations and identify areas for improvement.
- **Overcoming differences in understanding legislation:** Modeling helps transform legal expertise into actionable knowledge for non-lawyers and vice versa, promoting a transdisciplinary approach to law in public health. The architecture model is as understandable to IT professionals as it is to legislators and practitioners.
- **Supporting interdisciplinary collaboration:** Improved communication is essential for the effective collaboration necessary to provide safe care. Modeling identifies key phenomena and relationships, leading to process optimization. Multidisciplinary understanding is key to the further development of health information systems.
- **Ensuring consistency and reducing errors:** Clear visualization can reduce the risk of ambiguities in legislative proposals. This leads to more robust legislation that is better adapted to the realities of healthcare practice.

- **Use of artificial intelligence:** The two-way link between the graphical form (OPD – enabling simulation) and the textual form (OPL – facilitating connection to large language models) opens up new possibilities, particularly in the automated linking of existing and newly created legislation.

The formalization and modeling of legislative processes is gradually shifting lawmaking from an intuitive "art" based on experience to an exact concept based on algorithms and modern modeling tools.

Literature

- [1.] Knott RP, Merunka V, Polak J. Process modeling for object oriented analysis using BORM Object Behavioral Analysis. *Proceedings Fourth International Conference on Requirements Engineering ICRE 2000 (Cat No98TB100219)*. IEEE; 2000. pp. 7–16. doi:10.1109/ICRE.2000.855566
- [2.] Knott R, Merunka V, Polak J. The BORM methodology: a third-generation fully object-oriented methodology. *Knowledge-Based Systems*. 2003;16: 77–89. doi:10.1016/S0950-7051(02)00075-8
- [3.] Merunka V, Brozek J, Sebek M, Polak J. BORM-business object relation modeling. 2009. Available: <https://aisel.aisnet.org/amcis2009/788/>
- [4.] Brožek J, Merunka V, Merunková I. Organization modeling and simulation using BORM approach. *Lecture Notes in Business Information Processing*. Berlin, Heidelberg: Springer Berlin Heidelberg; 2010. pp. 27–40. doi:10.1007/978-3-642-15723-3_3
- [5.] Molhanec M, Merunka V. BORM: Agile Modelling for Business Intelligence. *Business Intelligence and Agile Methodologies for Knowledge-Based Organizations: Cross-Disciplinary Applications*. IGI Global; 2012. pp. 120–131. doi:10.4018/978-1-61350-050-7.ch006
- [6.] Merunka V, Pergl R, Tůma J. BORM-II and UML as accessibility process in knowledge and business modeling. *Lecture Notes in Electrical Engineering*. Cham: Springer International Publishing; 2015. pp. 1–6. doi:10.1007/978-3-319-06764-3_1
- [7.] Pavlicek J, Rod M, Pavlickova P. Usability evaluation of business process modeling standards – BPMN and BORM case study. *Lecture Notes in Business Information Processing*. Cham: Springer International Publishing; 2021. pp. 93–104. doi:10.1007/978-3-030-79022-6_9
- [8.] Craft Case. *Business Process Analysis*. In: *Visualize, define and simulate your processes. Locate inefficiencies, performance deviations or errors was never easier*. [Internet]. Available: <https://craftcase.com/>
- [9.] Berger J. Method and system for automated request modeling. US Patent. 7904431, 2011. Available: <https://patentimages.storage.googleapis.com/4f/3b/91/efd8c98d224638/US7904431.pdf>
- [10.] Kofránek J, Berger J, Polák J, Vojtěch A. Modeling ehealth processes using hierarchical state machines (statecharts). *Medsoft*. 2018;30: 35–56. Available: http://www.creativeconnections.cz/medsoft/2018/Medsoft_2018_Kofranek1.pdf
- [11.] Kofránek J, Berger J, Štěpánek P, Vrabel F, Vojtěch A. Modeling e-prescription processes. *Medsoft*. 2019;31: 57–65. Available: https://www.medsoft.website/sbornik/2019/Medsoft_2019_Kofranek_Print.pdf
- [12.] Bruthans J, Kofránek J, Vojtěch A. Concept and practice of electronic prescription. *Medsoft*. 2021;33: 89–92. Available: https://scholar.google.com/citations?view_op=view_citation&hl=en&citation_for_view=Qv8pd7MAAAAJ:0N-VGjzr574C
- [13.] Berger J, Bruthans J, Vojtěch A, Kofránek J. Using process model to define the legislative framework of electronic prescription in the Czech Republic. *Health Informatics J*. 2024;30: 1–12. doi:10.1177/14604582241270902
- [14.] Berger J, Bruthans J, Kofránek J. Improving Implementation of an Electronic Prescription System for COVID-19 Vaccination in the Czech Republic: Process Modeling Approach. *JMIR Formative Research*. 2023;7: e41575. Available: https://scholar.google.com/citations?view_op=view_citation&hl=en&citation_for_view=Qv8pd7MAAAAJ:axFR9aDTf0C

- [15.] Atkinson K. 2.1 – representation of legal information from A – information representation, preprocessing, and document assembly. In: Katz DM, Dolin R, Bommarito M, editors. *Legal informatics*. Cambridge University Press; 2021. pp. 35–40. doi:10.1017/9781316529683.004
- [16.] Bench-Capon T, Araszkiwicz M, Ashley K, Atkinson K, Bex F, Borges F, et al. A history of AI and Law in 50 papers: 25 years of the international conference on AI and Law. *Artif Intell Law*. 2012;20: 215–319. doi:10.1007/s10506-012-9131-x
- [17.] Michálek J. Modeling legal norms using computer programs. 2021. Available: <https://dspace.cuni.cz/handle/20.500.11956/170531>
- [18.] Šavelka J, Myška M, Ptašník A, Spáčilová D. *Legal information systems*. Brno: Tribun EU; 2011. Available: <https://scholar.google.com/citations?user=02Dvk88AAAAJ&hl=en&oi=sra>
- [19.] Waddington M. Rules as code: Drawing out the logic of legislation for drafters and computers. *SSRN Electron J*. 2022. doi:10.2139/ssrn.4299375
- [20.] Xscential. How to Construct RFPs and Tenders for Legislative Modernization. In: *LegisPro Standards** [Internet]. Available: <https://xscential.com/legis-standards>
- [21.] Lucke JV, Fitsilis F, Etscheid J. Using artificial intelligence for legislation – thinking about and selecting realistic topics. *EGOV-CeDEM-ePart**. 2022. Available: <https://dgsociety.org/wp-content/uploads/2022/09/CEUR-proceedings-2022.pdf#page=46>
- [22.] Hugo A. López, Søren Debois, Tijs Slaats & Thomas T. Hildebrandt. *Business Process Compliance Using Reference Models of Law*. 1st ed. In: Wehrheim H, Cabot J, editors. *Lecture Notes in Computer Science*. 1st ed. Cham, Switzerland: Springer Nature; 2020. doi:10.1007/978-3-030-45234-6_19
- [23.] Epifanova TV, Vovchenko NG, Toporov DA, Pozdnyshov AN. Development of legal education and machine-readable law in the conditions of economy digitization. *Digital Economy: Complexity and Variety vs Rationality*. Cham: Springer International Publishing; 2020. pp. 971–979. doi:10.1007/978-3-030-29586-8_110
- [24.] Flatt A, Langner A, Leps O. Model-driven development of akoma ntoso application profiles: A conceptual framework for model-based generation of XML subschemas. 1st ed. Cham, Switzerland: Springer International Publishing; 2023. doi:10.1007/978-3-031-14132-4
- [25.] OASIS. In: Akoma Ntoso Version 1.0 – OASIS Open [Internet]. 2018. Available: <https://www.oasis-open.org/standard/akn-v1-0/>
- [26.] MinBZK/leos: Legislation Editing Open Software – GitHub. Available: <https://github.com/MinBZK/leos>
- [27.] Yahya M, Breslin JG, Ali MI. *Semantic Web and Knowledge Graphs for Industry 4.0*. *Appl Sci (Basel)*. 2021;11: 5110. doi:10.3390/app1115110
- [28.] Pan JZ. *Resource Description Framework. Handbook on Ontologies*. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009. pp. 71–90. doi:10.1007/978-3-540-92673-3_3
- [29.] Antoniou G, van Harmelen F. *Web ontology language: OWL. Handbook on Ontologies*. Berlin, Heidelberg: Springer Berlin Heidelberg; 2009. pp. 91–110. doi:10.1007/978-3-540-92673-3_4
- [30.] Boella G, Caro LD, Humphreys L, Robaldo L, Rossi P, van der Torre L. Eunomos, a legal document and knowledge management system for the Web to provide relevant, reliable and up-to-date information on the law. *Artif Intell Law*. 2016;24: 245–283. doi:10.1007/s10506-016-9184-3
- [31.] Cvrček F. Legal electronic dictionary (PES). *Lawyer Year*. 2015;154: 247–260. Available: https://www.ilaw.cas.cz/upload/web/files/pravnik/issues/2015/3/4.INF_Cvrcek_3_2015.pdf
- [32.] Unhelkar B. *Software engineering with UML*. London, England: CRC Press; 2020. doi:10.1201/9781351235181
- [33.] Friedenthal S, Moore A, Steiner R. *A practical guide to SysML: the systems modeling language*. Morgan Kaufmann; 2014. Available: <https://www.academia.edu/download/73088918/c158c95430a561305e3e050e9ee74011e55b.pdf>
- [34.] Sparx System. In: *Enterprise Architect* [Internet]. Available: <https://sparxsystems.com/>
- [35.] Archi. In: *The Open Source modeling toolkit for creating ArchiMate models and sketches. Used by Enterprise Architects everywhere*. [Internet]. Available: <https://www.archimatetool.com/>
- [36.] Process modeling of public administration agendas. In: *Digital and Information Agency* [Internet]. Available: <https://www.dia.gov.cz/cs/nase-cinnosti/projekty/projekty-fondy-eu/procesni-modelovani-agend-verejne-spravy>
- [37.] Dori D, Tombre K. From engineering drawings to 3D CAD models: are we ready now? *Comput Aided Des*. 1995;27: 243–254. doi:10.1016/0010-4485(95)91134-7
- [38.] Dori D. Documenting system specifications through opm web-based graphics/text equivalence: the case of the free flight system. *J Logic Comput*. 1995;5: 227–249. Available: <https://dovdori.technion.ac.il/wp-content/uploads/2022/04/DocumentingSystemSpecificationsThroughOPM.pdf>
- [39.] Dori D. Document analysis systems development and representation through the object-process methodology. *Document Analysis Systems: Theory and Practice*. Berlin, Heidelberg: Springer Berlin Heidelberg; 1999. pp. 271–282. doi:10.1007/3-540-48172-9_22
- [40.] Dori D. *Object-process methodology: A holistic systems paradigm*. Berlin, Heidelberg: Springer Science & Business Media; 2002. doi:10.1007/978-3-642-56209-9
- [41.] Dori D. *Model-based systems engineering with OPM and SysML*. 1st ed. New York, NY: Springer; 2016. doi:10.1007/978-1-4939-3295-5
- [42.] Kang X, Shteingardt V, Wang Y, Dori D. Neuro-Conceptual Artificial Intelligence: Integrating OPM with deep learning to enhance question answering quality. *arXiv [cs.CL]*. 2025. Available: <http://arxiv.org/abs/2502.09658>
- [43.] Dori D. *Object-Process Methodology. Encyclopedia of Knowledge Management, Second Edition*. IGI Global; 2010. pp. 1208–1220. doi:10.4018/978-1-59904-931-1.ch116
- [44.] Dori D, Kohen H, Jbara A, Wengrowicz N, Lavi R, Soskin NL, et al. *OPCloud: An OPM integrated conceptual-executable modeling environment for industry 4.0. Systems Engineering in the Fourth Industrial Revolution*. Wiley; 2019. pp. 243–271. doi:10.1002/9781119513957.ch11
- [45.] Casebolt JM, Jbara A, Dori D. Business process improvement using Object-Process Methodology. *Syst Eng*. 2020;23: 36–48. doi:10.1002/sys.21499
- [46.] Vizitiu C, Dominey K, Nistorescu A, Dinulescu A, Marin M. MHealth case study presenting design SynMeth, a rapid prototyping MBSE methodology, by advancing specific OPM-to-SysML mapping. *IEEE Access*. 2025;13: 53531–53545. doi:10.1109/access.2025.3553945
- [47.] Dori D. Model-based standards authoring: ISO 15288 as a case in point. *Syst Eng*. 2024;27: 302–314. doi:10.1002/sys.21721
- [48.] ISO (the International Organization for Standardization). *Automation systems and integration – Object-Process Methodology*. 2024 Jan. Report No.: ISO 19450:2024. Available: <https://drive.google.com/file/d/1mgrRDQJ7BU5DPcGX6mdqWp8nnYLfXWcm/view?usp=sharing>
- [49.] Klievtsova N, Mangler J, Kampik T, Rinderle-Ma S. Utilizing process models in the requirements engineering process through Model2Text transformation. 2024 IEEE 32nd International Requirements Engineering Conference (RE). IEEE; 2024. pp. 205–217. doi:10.1109/re59067.2024.00028
- [50.] Deutsch H., Marshall O., Irwine A. D. Russell's paradox. In: Zalta and Uri Nodelman E, editor. *Stanford Encyclopedia of Philosophy (Winter 25 Edition)*. Stanford: The Metaphysics Research Lab, Department of Philosophy, Stanford University, Stanford, CA 94305; 2025. Available: <https://plato.stanford.edu/archives/win2024/entries/russell-paradox/>
- [51.] Biletzki A, Matar A. Ludwig Wittgenstein. In: Zalta and Uri Nodelman E, editor. *The Stanford Encyclopedia of Philosophy Fall 23 Edition*. Stanford: The Metaphysics Research Lab, Department of Philosophy, Stanford University, Stanford, CA 94305; 2023. Available: <https://plato.stanford.edu/archives/fall2023/entries/wittgenstein/>
- [52.] Hodges W. Tarski's truth definitions. In: Zalta and Uri Nodelman E, editor. *The Stanford Encyclopedia of Philosophy Winter 22 Edition*. Stanford: The Metaphysics Research Lab, Department of Philosophy, Stanford University, Stanford, CA 94305; 2022. Available: <https://plato.stanford.edu/archives/win2022/entries/tarski-truth/>

[53.] García Alarcia RM, Russo P, Renga A, Golkar A. Bringing systems engineering models to large language models: An integration of OPM with an LLM for design assistants. *Proceedings of the 12th International Conference on Model-Based Software and Systems Engineering*. SCITEPRESS – Science and Technology Publications; 2024. doi:10.5220/0012621900003645

Contact

Mgr. et Mgr. Adam Vojtěch, MHA
Biocybernetics Laboratory
Institute of Pathological Physiology,
First Faculty of Medicine,
Charles University
U nemocnice 5
128 53 Prague 2
adam.vojtech@lf1.cuni.cz

doc MUDr. Jiří Kofránek, CSc.
Biocybernetics Laboratory
Institute of Pathological Physiology,
First Faculty of Medicine,
Charles University
U nemocnice 5
128 53 Prague 2
jiri.kofranek@lf1.cuni.cz
+420 777 686868

doc. Ing. Vojtěch Merunka, Ph.D.
Faculty of Economics and Man-
agement, Czech University of Life
Sciences Prague
Kamýcká 129
165 00 Prague-Suchdol
merunka@pef.czu.cz
<https://vojtech.merunka.eu/>
tel.: +420 224 382 272