UDC:004.43
Original scientific paper

MODEL-BASED VALIDATION AND VERIFICATION OF ANOMALIES IN LEGISLATION

Vjeran Strahonja
Faculty of Organization and Informatics, Varaždin, Croatia
vjeran.strahonja@foi.hr

Abstract: *An anomaly in legislation is absence of completeness, consistency and other desirable properties, caused by different semantic, syntactic or pragmatic reasons. In general, the detection of anomalies in legislation comprises validation and verification. The basic idea of research, as presented in this paper, is modelling legislation by capturing domain knowledge of legislation and specifying it in a generic way by using commonly agreed and understandable modelling concepts of the Unified Modelling Language (UML). Models of legislation enable to understand the system better, support the detection of anomalies and help to improve the quality of legislation by validation and verification. By implementing model-based approach, the object of validation and verification moves from legislation to its model. The business domain of legislation has two distinct aspects: a structural or static aspect (functionality, business data etc.), and a behavioural or dynamic part (states, transitions, activities, sequences etc.). Because anomalism can occur on two different levels, on the level of a model, or on the level of legislation itself, a framework for validation and verification of legal regulation and its model is discussed. The presented framework includes some significant types of semantic and syntactic anomalies. Some ideas for assessment of pragmatic anomalies of models were found in the field of software quality metrics. Thus pragmatic features and attributes can be determined that could be relevant for evaluation purposes of models. Based on analogue standards for the evaluation of software, a qualitative and quantitative scale can be applied to determine the value of some feature for a specific model.*

Keywords: *Modelling legislation; UML business models, validation and verification, anomalies in legislation.*

1. INTRODUCTION – MODELLING OF LEGISLATION

The term "legislation" in this paper refers to the set of laws, statutes and other legal acts that cover a particular subject of law or practice.

Modelling is an essential part of business analysis and reengineering, as well as of software development. Specific modelling methods and techniques are enabling specification, visualization, and documentation of business and system models. Models and domain knowledge they contain may be shared, discussed and reused across groups of stakeholders and implemented in computer applications.

Some advantages of modelling may be used in domain of legislation. By implementing model-based approach, we used the advantage to move the object of validation and

verification from legislation to its model. Models of legislation are different in their purpose, level of abstraction and applied concepts. Models of legislation enable to understand the system better, support the detection of anomalies and help to improve the quality of legislation by validation and verification.

1.1. BUSINESS, PROCESS AND SYSTEM MODELS

Although the leading idea today is using the same modelling concepts and language for business, (business) process and software (system) modelling, these terms should be clarified.

A generally accepted distinction between business and system modelling is that a business modelling discusses how a business responds to a stakeholders or an event, whereas a system modelling deals the software and other information and communication technologies.

The goal of business modelling is to reach a common understanding between stakeholders regarding *who* is offering and exchanging *what* (goods, services, value) with *whom* and expects *what* in return [3]. The goal of a business process model is to specify *how* and by *whom* processes are carried out. Business modelling is centred around the notion of *value*, while in business process modelling concepts focus on *how* a process should be carried out.

Business modelling of legislation focuses on the substantive aspects of legislation, and business process modelling on the procedural aspects of legislation. The separation of substantive and procedural aspects of legislation is well known. The procedural regulation defines the "court procedure" in terms of the process that the case will go through. From the point of view of parties and judge, procedural regulation comprises the rules for proceedings the enforcement of substantive law that will occur in different situations. Application of the procedural regulation is not focused on the quality of substantial decisions, but on the quality of the process (workflow, duration, delays, number of hearings etc.). In contrast to procedural, the substantive regulation (i.e. law or its part) deals with the "substance" of the matter. It defines how the facts in some type of the case or legal procedure will be handled, how the crime will be charged, or the dispute will be resolved. Simply, the substantive law defines crimes and punishments. The substantive regulation focuses on quality of court decisions.

Software (system) modelling is a structured way of applying the modelling approach to the business itself, designing software requirements and other models for the subsequent software design activity. The motivation for development of system models of legislation is mostly a desire to build a court case index, document management system or case management system.

1.2. STRUCTURAL AND BEHAVIOURAL MODELLING WITH UML

From the point of view of modelling, a business domain has two distinct aspects: a structural or static aspect (functionality, business data etc.), and a behavioural or dynamic part (states, transitions, activities, sequences etc.). From the point of view of this paper, emphasis is on the behavioural features of a system, e.g. the ways a system behaves in response to certain events or actions.

The basic idea of modelling law, as presented in this paper, is capturing domain knowledge of procedural legislation and specifying it in a generic way by using commonly agreed and understandable modelling concepts of the Unified Modelling Language (UML) [8]. Currently, UML is de facto standard for expressing object-oriented analysis, design modelling and documenting object-oriented and component-based system architectures. Although

the strengths of UML are at software development, it is commonly used for representing business domain. UML models of legislation provide a framework for validation and verification of legal regulation and its model.

UML models offer over all:

- describing system structure and behaviour in an intuitive way by using visual modelling
- readability and understandability by other human readers, and lower level of required expertise, as compared with formal specifications
- lower ambiguity, as compared with natural languages.

UML is divided into structural and behavioural specifications, i.e. models of the static and dynamic aspects of a system.

Structural models represent the overall object structure of the business domain or of the software system. On the conceptual and logical level, they provide the static representation of the business domain and/or software system in terms classes, actors and use cases. Although the static aspects of legislation are also very important, this paper focuses on behavioural modelling of legislation. More specific, this paper presents an analysis approach based on the UML state machine diagrams of legislation. Ideally, we would like to have such model of legislation, even formal specifications, to check correctness and consistency of legal regulation and its model.

Behavioural models represent different aspects of dynamic behaviour, i.e. how the structural aspects of a system change over the time. Behavioural models of legislation, in a form of UML diagrams, provide a graphical notation for describing the dynamic (time-dependent) behaviour of a legal system and improve understanding of a legal domain. They focus on the object states and events causing changes of object states, including message-passing between objects, sequence and conditions for invoking other behaviours. On the conceptual and logical level, UML has three behaviour diagrams: activity, state machine, and sequence. Each kind of behavioural model focuses a different aspect of business or system dynamics. It makes one or the other diagram more suitable for a particular stage of application development or application domain.

1.3. SYNTAX, SEMANTICS AND PRAGMATICS OF MODELS

Most of the theories, methods and formal approaches in a field of business modelling and modelling of legislation originated in linguistics and in knowledge based systems. Some general aspects are similar in both linguistics and modelling theory, like syntax, semantics and pragmatics.

As an analogue of linguistics, we can also define syntactic, semantic and pragmatic anomalies of models. Syntax prescribes in the natural language the way in which words and phrases are combined to form sentences (the deep and structure of sentences). In the case of models, modelling syntax comprises the set of allowed modelling concepts, reserved words and their parameters and the correct way in which modelling concepts are used. Syntactic anomalies are caused by a violation of the structural (grammatical) rules for the modelling technique.

Semantics is a field of linguistics defined as the study of meaning of words, phrases, sentences, and texts. In a modelling theory semantics deals with the meaning systems of modelling language and concepts and their mapping to the real world. Semantic anomalies deal with violation of meaning and sense, for example conflicting truth conditions, name

conflicts, dangling references etc. In a modelling theory, semantic anomalies are mostly result of inconsistent or inadequate use of modelling concepts.

Pragmatics is the study of information structure and the use of language in communicative context. Pragmatics is concerned with bridging the gap between a theory and its implementation in some context.

2. ANOMALIES IN LEGISLATION

The most desirable "technical" properties of legislation, but not all, are completeness, consistency and logical/semantic contradiction. Absence of this and other desirable properties are anomalies in legislation. It's generally accepted that anomalies in legislation impact on the implementation and enforcement of law.

Some other theories that we need for validation and verification of legislation and its models are traditionally addressed in knowledge based systems. Although research of anomalies in legislation is still an attractive field for research, we use some common issues like consistency, completeness and logical/semantic contradiction, that are considered more then ten years ago [1, 12].

Incompleteness is the failure of completeness. Generally, there is at least one improvable schema (sentence, statement) that could be added as an axiom schema without creating simple inconsistency. Incompleteness issues are: dead-end rules, missing rules, unreachable rules, dangling references, unreferenced attribute values and other unintentional non-determinism.

Inconsistency of the specification implies that there are conflicting statements. Discrepancy is simply the difference between conflicting statements, definitions or rules of the same fact or situation. Some of the appearances of inconsistency are redundancy, unnecessary IF conditions logical contradiction, subsumed rules, circular rule, name conflicts (synonyms, homonyms), inconsistent generalization/specialization and other logical/semantic contradictions.

Anomalies in legislation may be caused by different semantic, syntactic or pragmatic reasons. For example, homonyms are pure semantic anomalies (if they are not desired), but an unintentional non-determinism can occur in the model as a semantic anomalism of the legal pattern, or a syntactic failure.

Anomalism can occur on two different levels, on the level of a model, or on the level of legislation itself. For example, some missing rule can disappear during a modelling process, but can also be omitted in regulation during the legislative procedure.

Since manual checking of anomalies in legislation is error-prone and time-consuming, currently the development of computer supported and automated methods for validation and verification of legislation attract researchers and practitioners from all around the world. All these validation and verification methods lie on decreasing complexity of legislation, by using some methods of modelling laws.

Pragmatics is a discipline of connecting a theory and its implementation. It is the same in the linguistics and in the modelling theory, but research of practical anomalies of models seems to be of minor interest as compared with linguistics. In linguistics, pragmatic anomalies deal with a discrepancy between literal meaning of the sentence and the speaker's meaning in the context of conversation.

Some ideas for assessment of pragmatic anomalies of models were found in the field of software quality metrics. As presented in Table 1., this is a quantitative scale and method which can be used to determine the value of some feature for a specific software product and ISO 9126 is an international standard for the evaluation of software [4].

Table 1: Pragmatic features and attributes relevant for evaluation purposes of models

Pragmatic feature	Attribute
Functionality – the existence of a set of functions that satisfy stated or implied needs.	Suitability for specified tasks or class of problems
Usability – the effort needed for use, and the individual assessment of such use.	Understandability of modelling concepts and models
Efficiency – the relationship between the level of performance of the model and amount of resources needed to build it	Amount of resources used and the duration of such use for modelling and activities
Maintainability - the effort needed to make specified modifications	- Analysability of model in terms of the effort needed for diagnosis of deficiencies, anomalies and for identification of parts to be modified - Changeability in terms of the effort needed for modification and fault removal - Testability in terms of the effort needed for validation and verification of the modified models

3. MODELLING AND ANALYSIS OF LEGISLATION

Based on current research, we suggest the iterative model-based approach to the validation and verification of legislation. The assumption of this approach is that the domain expert can't create the model himself, but the modelling engineer is at least the moderator of this modelling process.

The modelling process of legislation consists of four steps (Fig. 1).

- Basic analysis of selected legislation (classification, conceptualization and refactoring)
- Transformation to UML constructs and representation in a form of diagrams (interpretation, formalization)
- Validation and verification (detection of anomalies based on static and dynamic analysis)
- Improvement of model and legal sources.

Basic analysis of selected legislation comprises classification, conceptualization and refactoring of legislation. The knowledge on legislation is described in natural language. In this phase, the legal sources and additional knowledge gained from the domain expert, which is represented in natural language, must be interpreted and structured. Within this step a gap between sometimes unstructured and semiformal descriptions of the legal expertise has to be bridged. Classification of legal statements must take into account some classification patterns (procedural-substantial, terms and definitions, case management, court activity, making decision, conducting the procedure, document management, communication …). Conceptualization comprises the identification of structural and behavioural constructs of selected legal act. Refactoring is the process of rewriting of legal source to improve its readability and structure from the point of view of further modelling technique, with the explicit purpose of keeping the meaning and behaviour of the source. The applied refactor-

ing form was simple table, with columns who/actor, facts and rules (time limit, initial state, event, action, final state).

Transformation to UML constructs and representation in a form of diagrams comprises interpretation and formalization. The outputs from this step are structural and behavioural models. As described previously, these closely related models represent the same legal domain in a different way and in another representation. To gain the full benefits for validation and verification, these different models have to be interrelated explicitly.

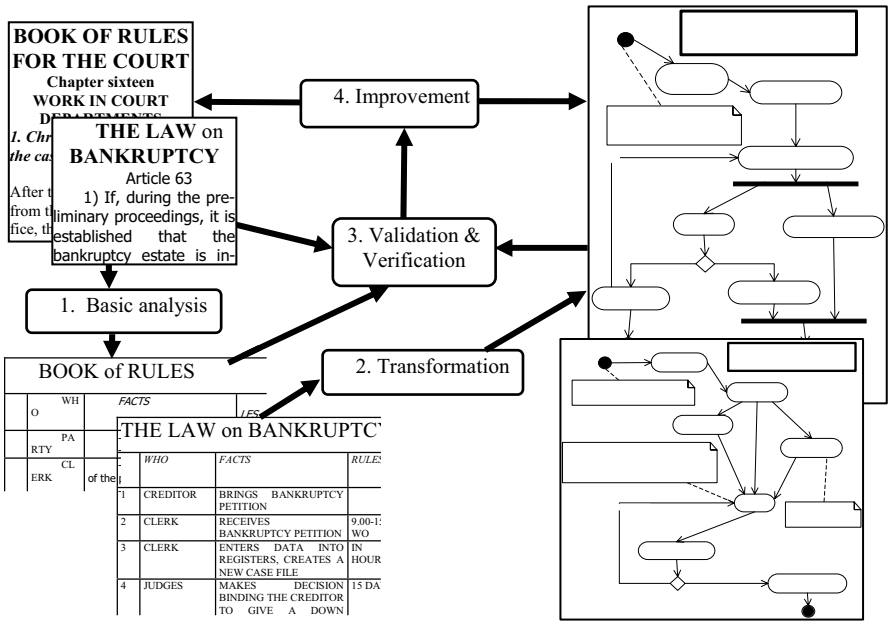

Figure 1: Iterative process of modeling

4. VALIDATION AND VERIFICATION

From the point of view of this paper, anomalism can occur on two different levels, on the level of a model, or on the level of legislation itself. From this point of view a framework for validation and verification of legal regulation and its model is discussed.

In general, the detection of anomalies in legislation comprises validation and verification (V&V). Although the objective is the same, the approaches to validation and verification differ in their orientation. In the field of software engineering, validation answers the question: "Are we building the right product?", and verification: "Are we building the product right?" [2]. Validation means testing some model or specification against the users' requirements and expectations, and verification means testing against the design specification, methodology, use and constrains of modelling concepts, rules of design etc.

Validation is the process of checking if statements of some legal act are true, if it works as intended, if it meets common regulatory requirements and statutory compliance that may be very fuzzy, changeable and ambiguous. Validation is mostly based on human expert opinion. The idea of validation, as applied in this paper, is to transform legal structure and procedure into a visualization model, to enable experts to validate their scenarios. The validation of legislation and determination of anomalies can not be automated, but visualization of legal structure and procedures can help the expert to make decision whether some poten-

tial anomaly (redundancy, synonym, circular definition etc.) is really an anomaly in legislation or not.

Validation should not be confused with verification. Verification is the act of proving or disproving the correctness of a legal act with respect to a certain specification or property. Verification is the process of reviewing, auditing, inspecting, testing, checking, or otherwise establishing and documenting whether some specification or model conforms to previously determined requirement. In contrast of validation, that is a human-directed proof, verification can be automated to some extend, as described in relevant papers [7, 5, 6, 9, 10, 11].

As UML and its modelling techniques move from academic institutions into commercial software development and domain modelling, they have to fulfil stronger requirements concerning correctness and consistency, Therefore, verification and validation are inherent activities of modelling. There are two basic complementary analysis techniques for modelling legislation, static and dynamic [6]. These are compared in Table 1.

The static analysis lies on the concept of class and gives a behavioural model that is valid for all possible case proceedings. In our example, the Procedural Manual (the Book of Rules) defines a general court procedure, valid for all courts and all types of case. The Bankruptcy Law, the Law on Civil Proceedings, the Execution Law etc. define specific court procedure valid for all courts and for some specific type of case. Static analysis as a process comprises modelling and evaluation of a model or system, based on its form, structure, content, or documentation. The idea is to understand how the system works and establish certain correctness criteria. This in a conservative technique, where we analyze the implementation to prove which states and transitions are illegal. The static analysis checks those criteria that are not related with the global state space (an upper bound).

The dynamic analysis uses a specialization, i.e. an implementation sub model of the static analysis model that is valid for one particular case proceeding. Conceptually, it lies on the object as an implementation of the class. In the dynamic analysis, we observe instances of states and transitions that are a subset (a lower bound) of the ideal, complete model. We use these specializations as proof of existence. For example, reachability analysis is detection of unreachable states, undesired global states or illegal sequence of actions. Unfortunately, the examination of a global state space often results in a state space explosion.

Table 2: Comparison of static and dynamic analysis

Static Analysis	Dynamic Analysis
Represents all possible states and transitions in all possible case proceedings	Represents one particular case proceeding
Superset of ideal model, generalization, upper bound	Subset of ideal model, specialization, lower bound
Conservative analysis detects illegal states and transitions	Proof of existence and reachability analysis detects legal states and transitions
Assumes that an exception will be produced on an illegal input	Global state of space results in state space explosion

5. CONCLUSION

The development of a model-based approach to the validation and verification of legislation by using UML was partially motivated by experiences gathered during the development project of the Croatian Court Case Management System. During the phase of project

preparation (2000-2003), business models of current legislation, court proceedings and case management were developed. It consists of business use-case model, domain class model and behavioural statecharts/activity models. Statecharts were developed for general case management procedures, as well as for specific bankruptcy, enforcement, litigation and criminal procedures.

This was the prerequisite of system modelling and development phase (started in 2005), that focuses on different aspects of the computer system, such as programs that automate the business process and business rules, database, user interface, system procedures etc. During this phase statecharts are refined and converted to state machine notation.

Based on empirical research, assessment of used method is made.

Some improvements of methodology, like semi-automated syntactical verification are promising but require further research. Other field of further research are anomalies in legislation. Different types of anomalies in legislation are still classified and worked out, but this domain requires serious ontological research.

Last but not least, UML seems to be a cure-all with clearly described semantic concepts, standard notations and suggestions for implementation. But application of UML in particular domains, such as modelling of legislation, needs to be researched and evaluated.

REFERENCES

[1] Ayel M, Laurent JP (eds) (1991) Validation, Verification and Testing of Knowledge-Based Systems. John Wiley & Sons Ltd., Chichester, England

[2] Boehm B (1984) Verifying and Validating Software Requirements and Design Specifications. IEEE Software, vol 1, pp 75-88

[3] Decker S, Erdmann M, Studer R (1996) A unifying view on business process modelling and knowledge engineering. Proceedings of the 10th Knowledge Acquisition for Knowledge-Based Systems Workshop, Banff, Canada, pp. 34/1-34/16

[4] ISO (1991) ISO/IEC: 9126 Information technology - Software Product Evaluation - Quality characteristics and guidelines for their use. International Organization for Standardisation

[5] Kracht D, de Vey Mestdagh CNJ, Svensson JS (eds.) (1990) Legal Knowledge based systems, an overview for validation and practical use. JURIX '90. Vermande, Lelystad

[6] Latella D, Majzik I, Massink M (999) Towards a Formal Operational Semantics of UML Statechart Diagrams. In P. Ciancarini and R. Gorrieri, editors, IFIP TC6/WG6.1 Third International Conference on Formal Methods for Open Object-Oriented Distributed Systems, Kluwer Academic Publishers

[7] Plaza E (ed.) (1993) Validation & Verification of Knowledge- Based Systems. IEEE Expert. Special Issue , vol 3

[8] Rumbaugh J, Jacobson I, Booch G (1999) The Unified Modelling Language Reference Manual. Addison-Wesley

[9] Van Engers TM, Glassée EJJ (2001) Facilitating the Legislation Process Using a Shared Conceptual Model. IEEE Intelligent Systems, January/February , pp 50-58

[10] Van Engers TM, Kordelaar PJM, Ter Horst EA (2001) POWER to the E-Government. Knowledge Management in e-Government 2001, IFIP, ISBN 3 85487 246 1

[11] Van Engers TM, Gerrits R, Boekenoogen M, Glassée EJJ, Kordelaar PJM, (2001) POWER: Using UML/OCL for modelling Legislation - an application report. Proceed-

ings of the International Conference on Artificial Intelligence and Law, ACM 1-58113-368-5/01/0005

[12] Vermeasan A, Coenen F (1999) Validation and verification of knowledge based systems (theory, tools and practice). Knluwer Academic publishers