Modeling and Verification of Judicial Workflows

Malte Hübner[1], Hendrik Schöttle[2]
[1] Max-Planck-Institut für Informatik
[2] Institut für Rechtsinformatik, Universität des Saarlandes

1 Introduction

As a consequence of the European integration process, EU member states increasingly have to cooperate with foreign authorities, for instance *to serve a claim, to execute a judgment* or to *arrest a suspect*. These procedures necessitate firstly the knowledge of the respective parts of the foreign law and secondly communication with the authorities involved in the given case. The large number of differing legal systems in Europe implies that the vast majority of legal experts does not possess the relevant specialized knowledge.

Within the scope of the *eJustice* project we aim at developing a language in which knowledge about the aforementioned procedures can be encoded. This knowledge can then be made available to all involved parties in form of an *assistance system* and will also be used for the *partial automation of transnational judicial workflows*.

This question is related to traditional Business Process Management, but exhibits several specificities: While Business Process Management aims at describing and optimizing existing workflows, in our scenario also *potential* workflows need to be encoded which have to be extracted from the body of legislation. Legal regulations can thus be seen as a *specification* of judicial workflows. Computer-aided tools are needed to automatically check that a given model of a workflow meets the *specification* (i.e. the law).

2 Modeling Language

Even though our system features some characteristics of an expert system, we base our work on the hypothesis that the development of a complete expert system is not possible because the body of legislation is logically inconsistent. Therefore, the objective of our project is to model certain parts of mutual legal assistance. This results in the following requirements for an appropriate modeling language:

Refinement and modular specification A model of multi-national legal relations cannot be created and maintained by a single expert. We therefore start out with a rough model which can later be refined by experts working independently on different parts of it. The modeling language to be used in the project therefore has to support *hierarchical*

refinement as well as *modularization*.

Formal Semantics and Model Checking The description of judicial workflows requires a method to show that a given model is in line with the law. This is particularly difficult for extensive models which have been created by different persons. It might also be useful to be able to verify common *Safety- and Liveness-* properties, like *Every Workflow terminates in finite time*, etc. To automate this verification on a computer, we aim at applying the well-known method of *Model Checking* [CGP99]. Of course, this requires the definition of a formal semantics for the modeling language.

Choosing an appropriate specification language and a corresponding semantics necessitates to identify which types of properties need to be verified as valid for our models. Examples are: (i) "Every claim will *eventually* be decided." (ii) "Every *vorbereitender Schriftsatz* [...] is delivered *at least one week* before the hearing." (Deadline, quantification over documents) (iii) "Every document of *type* T is only processed by a *judge*." (Roles)

3 Judicial workflows and UML

After having laid out the requirements that a language for the modeling of judicial workflows must fulfill (hierarchical, modular, formal semantics) the question is what language to use. At the moment we are evaluating whether the *Unified Modeling Language* (UML) – possibly with some extensions – is suitable for our needs. Making use of these languages could turn out to be useful with respect for the following reasons: (i) UML is a well-known and well-documented language. Parts of UML are not solely used for software-design any more, but also in different areas like workflow modeling and the modeling of biological systems (e.g. [KCH01]). (ii) UML consists of a family of languages which makes it possible to model different aspects of a workflow in a single framework. (iii) Formal Semantics and *Model Checking* have already been suggested for formalisms which have now been integrated into UML (e.g. Eshuis [Es02] for Activity-Charts and Harel [Ha87] for Statecharts). Some concepts of UML such as Activity- and Statecharts provide the possibility for hierarchical refinement (cf. Sec. 2).

Literatur

[CGP99] Clarke, E. M., Grumberg, O., und Peled, D. A.: *Model checking*. MIT Press. 1999.

[Es02] Eshuis, H.: *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*. PhD thesis. University Twente. 2002.

[Ha87] Harel, D., Pnueli, A., Schmidt, J., und Sherman, R.: On the formal semantics of statecharts. In: *Proceedings of the Symposium of Logic in Computer Science*. S. 54–64. 1987.

[KCH01] Kam, N., Cohen, I., und Harel, D. The immune system as a reactive system: modeling t cell activation with statecharts. 2001.