Conceptual Models Supporting Formal Policy Modelling: Metamodel and Approach

Sabrina SCHERERand Maria A. WIMMER
University of Koblenz-Landau, Koblenz, Germany

Abstract. A novel approach for engaging stakeholders in policy making is developed and implemented in the OCOPOMO project. Scenario texts generated by stakeholders as well as other background documents are used to inform formal policy modelling, which applies declarative rule-based agent modelling paradigms. To bridge the gap between narrative texts of scenarios and formal policy models, the OCOPOMO approach introduces the Consistent Conceptual Description (CCD) as a "modelling middle-layer". This paper presents the CCD metamodel, i.e. the underlying vocabulary to describe policy contexts, and it describes how the CCD metamodel supports the semi-automatic transformation of conceptual models of a policy context to generate formal policy models.

Keywords. Conceptual Models, Metamodel, Model Driven Development, Agent-Based Policy Modelling and Simulation

1. Introduction

In the OCOPOMO[1] project, a novel approach for engaging stakeholders in policy development is conceptualized and implemented [8]. Stakeholders are collaboratively involved in the development of scenario texts relevant in the context of a policy under discussion. By policy we refer to strategic areas of complex decision-making with various stakeholders having potentially diverging interests. In OCOPOMO, public policies are investigated and modelled such as renewable energy policy of the Kosice region in Slovakia, housing policy of the city of London or the distribution of structural funds in the Campania region in Italy. The overall OCOPOMO policy development process consists of six phases, where the first two phases involve stakeholders to generate scenarios of potential policy aspects, which are complemented with background documents to evidence statements in the scenarios. In phases three and four, the policy case is conceptualized and modelled by modelling experts, while phase five runs simulations to generate outcomes. In phase six, the results of the simulation are exposed to the stakeholders, who compare their scenarios (of phase 2) and the simulation outcomes in order to either update their scenarios or accept the insights from the simulation and agree that these are consistent with the inputs the stakeholders provided in the scenario generation phase. For more details on the OCOPOMO policy development process the reader is referred to [8].

[1] Open Collaboration Policy Modelling, http://www.ocopomo.eu/

As indicated before, the scenarios as well as other background documents inform formal policy models. The formal policy models are developed applying declarative rule-based agent modelling paradigms [5]. The main aims of formal policy modelling are to evaluate consistency of the simulation, to elicit and generate insights on the policy case which could not be detected solely from the narrative and conceptual models, and therewith to inform decision makers with a consistent formal and narrative representation of a potential policy to be decided. The overall OCOPOMO approach thereby ensures participation and involvement of stakeholders, and collaborative policy development. It increases transparency by making complex policy cases more understandable through (i) a set of artefacts representing distinct policy options, and (ii) enabling traceability of arguments put forward in the scenarios or reported in background documents and embodied in formal policy models.

OCOPOMO requires a multi-disciplinary approach for eliciting opinions and viewpoints of stakeholders, for analysing and mining large text sources, for conceptualizing the policy context, and for developing and running simulation models. The project therewith faces two challenges as outlined in [3]: "In individual developments, it is difficult that one person has all the expertise required; in teams, there are communication problems". In order to solve or reduce communication and integration challenges, "the use of agent-oriented software engineering methods and tools, relying on a modelling middle-layer" is proposed [3]. To tackle these challenges and to bridge the gap between narrative texts of stakeholder-generated scenarios (evidenced through background documents of the policy to be discussed) and formal policy models (generating model-based scenarios), the OCOPOMO approach introduces the Consistent Conceptual Description (CCD) as "modelling middle-layer". A CCD enables different stakeholders in the OCOPOMO process to better understand the policy context and to support semi-automatic transformation of text statements into formal statements and agent descriptions. Thereby, the CCD is exposed to the following needs and requirements:

Consistent By relying on visualization and ontological structures, the CCD aims to make modelling decisions understandable and traceable for stakeholders. In the CCD, text phrases from source scenarios and background texts are linked with the conceptual descriptions. Semi-automatic transformation of a CCD into imperative and declarative programming code (thereby also conveying references to scenarios and background documents) allows that simulation results can be linked with original text phrases. Through the conceptualization, the CCD supports in developing consistent policy models.

Conceptual The CCD aims to provide a conceptual model of a policy case i.e. conceptualising actors, policies, beliefs, aims etc. and their relations relevant in a policy case, which are described in scenarios and background documents. Concepts are further transferred into concrete programming code elements based on a Model Driven Development (MDD) approach known from software engineering in agent-based simulation and modelling as e.g. described in [1,4].

Description Along the transformation to formal policy models, the CCD plays a role of an intermediary between scenarios and simulation models. Several scenarios can form input to the CCD of a policy domain and further lead to a formal simulation model. Likewise, expertise of policy analysts may lead to particular knowledge constructs in the CCD. The CCD may also inform the scenario development of stakeholders by visualising particular knowledge gaps in the existing scenario descriptions. Finally, the CCD content may be revised or enriched based on input from analysing simulation results (model-

based scenarios, i.e. interpretations of simulation outcomes, and supportive graphical representations of model outcomes). [8]

In this paper, we present the CCD metamodel, i.e. the underlying vocabulary and relations to describe policy contexts. We further sketch the concept for semi-automatic transformation of conceptual policy descriptions of the CCD into formal policy models. The remainder of the paper is as follows: Section 2 describes the CCD metamodel. Section 3 introduces related work and highlights the differences of the CCD metamodel from these concepts. In Section 4, the technical implementation to support the OCOPOMO process in bridging narrative text with formal policy models via the CCD tool is briefly introduced. The paper concludes with a brief reflection of the added value of the approach and needs for further research.

2. Metamodel for the Consistent Conceptual Description (CCD-Metamodel)

A CCD represents the conceptualisation of a policy case. To ensure well-defined and structured CCD models of a policy domain, the CCD metamodel serves as a blueprint or vocabulary for describing a policy case with the requirements mentioned in Section 1. Formally, a metamodel can be defined as a conceptual description of a modelling, which illustrates the model concepts as well as their usage [9]. Thereby the CCD metamodel defines a "set of representational primitives with which to model a domain of knowledge or discourse" as typically an ontology does [2]. Representational primitives of an ontology are typically "classes (or sets), attributes (or properties), and relationships (or relations among class members)" [2]. Based on the approach for ontology of [2] and the rationale of metamodels according to [9], a set of elements of the CCD metamodel are defined. The core elements for the CCD metamodel are derived from insights into the three policy cases in OCOPOMO as well as from discussions with policy modellers in the project. The following (static) elements have been defined for the CCD metamodel: *Concepts* with *Actors*, *Objects* (representing particular entities or sets), *Attributes* and *Relations*. Facts for the simulation model are encoded with *Instances* of a concept *Actor* or *Object*. Dynamic aspects are encoded as *Action*. Each concrete *Action* has an actor, inputs and outputs (*ActionInputOutput*). This allows to model behaviour and interaction in one model. All elements can be linked with each other. Figure 1 shows the CCD metamodel with its core elements. The elements are further described in Table 1. In addition to the core elements of the CCD metamodel, the class *Annotation* links a concept or concrete instance of the CCD with its basic information in a scenario or background document. Two types of annotations are possible: text annotations and expert annotations. An annotation consists of a URL to the document referenced, a start position and the length of the phrase forming the annotation. The annotation is directly stored in the CCD.

3. Model Driven Development For Agent-Based Modelling and Simulation

Approaches applying MDD in agent-based modelling are e.g. described in [7,1,4,6]. The different approaches use different vocabularies / metamodels for conceptual models and transformation procedures. The annotation component in OCOPOMO differentiates the CCD metamodel from those metamodels found in literature. The CCD metamodel fol-

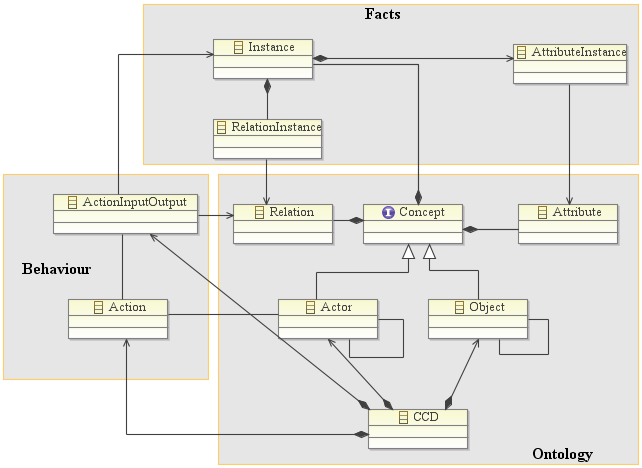

Figure 1. The CCD metamodel

Table 1. Elements of the CCD metamodel

Element	Description
CCD	Representation of the environment with its actors, objects, and actions.
Concept	Concept is an abstract class. A Concept can have instances, relations and attributes. There are two concrete concepts available: Actor and Object
Actor	Representation of a class of actors in the policy case - i. e. the agents in the multi-agent based simulation model. Actors are active and perform pro-active actions. Actions executed by a class of actors can be linked with the actor class.
Object	Representation of a class of objects in the policy case, i. e. all objects that are passive. Hence actions are not linked with an object class. However, objects can be part of an ActionInputOutput object and therefore be changed by an action - such an action is called a reaction.
Relation	Representation of relationships between two actors, two objects, or an actor and an object.
Attribute	Representation of attributes/properties for a class of actors or objects.
Instance	Representation of a concrete occurrence of an actor or object class. Instances have concrete attribute values (AttributeInstance) or have relationships (RelationInstance) with other instances. Instances define facts.
Action	Representation of tasks or actions in the policy case. Pro-active actions are executed by actors and pursue an objective. Reactions have no actor defined. Each Action can have several inputs and several outputs represented by ActionInputOutput objects. ActionInputOutput represents an input or output of an action. An ActionInputOutput as output of one action can be used as input for another action. This way, behaviour of and interaction between actors are modelled. This construct also provides the basis for the rule-dependency graph in the declarative agent-based modelling.

lows a rather descriptive and ontology-based approach than most of the metamodels in literature. In addition to actors (agents), the CCD metamodel allows the definition of objects and instances and their relations; however, without specifying the types of objects - i.e. the conceptual modeller has freedom. The freedom of the modeller makes the transformation from conceptual descriptions into programming code more complex. Yet, the descriptive character of the conceptual model is seen by the OCOPOMO project partners as more important than the automatic code transformation (i.e. a semi-automatic transformation is accepted). From a conceptual viewpoint, the transformation of concep-

tual models into simulation models is done by building analogies between elements of the CCD metamodel and elements of the simulation model (the declarative rule-based agent modelling system (DRAMS) as introduced in [5]). Thereby a kind of transformation model is built, which is dependent on the CCD and DRAMS metamodels. The CCD metamodel and DRAMS metamodel themselves are independent from each other. The transformation model allows the specification of details relevant for the simulation model already in the conceptual description. To bridge the conceptual models and the formal simulation models, OCOPOMO foresees the following transformation steps: 1. Transformation of text into CCD, 2. Transformation of conceptual models (CCD) into conceptual simulation models 3. Transformation of conceptual simulation models into programming code.

4. Technical Implementation

The CCD Tool supports Facilitators and Policy Modellers in developing a stakeholder-accessible formalisation of a policy model and thereby ensuring and documenting the consistency via the CCD. It supports the following main functions: a) Creating, editing and saving a consistent conceptual description (CCD) for a policy case, b) Linking the concepts of a CCD with background documents and scenarios, and c) Creating formal policy models that can be further programmed in the Simulation Engine. The CCD Tool[2] as part of the OCOPOMO ICT toolbox has a modular design with the following components:

Conceptual Description (CD) Tool: supports the creation, editing and storing of a CCD file (XML) which is representing the conceptual description of a policy case. The structure of the file is defined in the CCD metamodel.

Annotation Tool: is used to annotate and link relevant text phrases from background documents and stakeholder scenarios with relevant elements in a CCD file.

Transformation Tool (CCD2DRAMS Tool): supports (draft) source code generation from a CCD file for the Simulation Model in DRAMS. Thereby the CCD2DRAMS Tool supports traceability of simulation results by linking code fragments with concepts of the underlying CCD file.

5. Conclusions and outlook

In this contribution, we have outlined the concept for bridging narrative text artefacts and formal policy models via a conceptual description to support policy modelling. We introduced the CCD metamodel and argued its differences from other meta-modelling concepts. Finally, we briefly outlined the implementation setup for the CCD tool. The main added value and benefits of the CCD tool (with the underlying CCD metamodel) are to enable non-expert policy developers to extract aspects of the policy domain under consideration, to conceptualize and structure the domain and to provide an understandable conceptual model of a policy domain, which can be used by expert policy model programmers to generate initial code fragments for the simulation model. A key added

[2]Implemented as Eclipse Plug-in using EMF (Eclipse Modeling Framework, http://www.eclipse.org/emf/) and GMP (Graphical Modeling Project, http://eclipse.org/modeling/gmp/

value for the stakeholders and the programmers is the concept of traceability, i.e. being able to trace an agent description, a fact description or rules firing in the simulation runs back to the descriptions from where these have been derived (the evidences). This way, the current request for open government with the principles of collaboration, openness and participation is supported to a large extent. The CCD tool (incl. the annotation, conceptual model development and transformation into DRAMS code) is currently tested along the three use cases of OCOPOMO. During the tests, the CCD metamodel is analysed and evaluated for different criteria: its comprehensibility by different project actors, its completeness and eligibility to model the policy cases, its practicability to generate formal policy models. The evaluation will result in revisions of the CCD metamodel presented in this paper. Further research is ongoing to implement the traceability which is to be enabled across different ICT tools, as the CCD toolbox builds on model-driven architecture. As this concept allows individual modules to be replaced by other software modules, further research will be needed to support adaptability and transformation of the tracing concept into other software modules that can be integrated in the OCOPOMO policy development process.

Acknowledgements

OCOPOMO is co-funded by the EC within FP 7, contract No. 248128. The authors acknowledge the contributions of and express their gratitude to the OCOPOMO project partners, especially Scott Moss, Klaus Troitzsch and Ulf Lotzmann. The content of this paper represents the view of the authors, respectively. The European Commission cannot be made liable for any content.

References

[1] J. Gómez-Sanz and J. Pavón. Agent oriented software engineering with INGENIAS. In *Proceedings of the 3rd Central and Eastern Europe Conference on Multiagent Systems, Springer Verlag, LNCS*, volume 2691, pages 394–403. Citeseer, 2003.

[2] T. Gruber. Ontology. In L. Liu and M. T. Özsu, editors, *Encyclopedia of Database Systems*. Springer-Verlag, 2009.

[3] S. Hassan, R. Fuentes-Fernández, J. Galán, A. López-Paredes, and J. Pavón. Reducing the Modeling Gap: On the use of metamodels in agent-based simulation. In *6th Conference of the European Social Simulation Association (ESSA 2009)*, pages 1–13, 2009.

[4] T. Iba, Y. Matsuzawa, and N. Aoyama. From conceptual models to simulation models: Model driven development of agent-based simulations. In *9th Workshop on Economics and Heterogeneous Interacting Agents*. Citeseer, 2004.

[5] U. Lotzmann and R. Meyer. DRAMS - A Declarative Rule-based Agent Modelling System. In *Proceedings of 25th European Conference on Modelling and Simulation*, 2011.

[6] F. Okuyama, R. Bordini, and A. da Rocha Costa. ELMS: An environment description language for multi-agent simulation. *Environments for Multi-Agent Systems*, pages 91–108, 2005.

[7] J. Pavón, J. Gómez-Sanz, and R. Fuentes. Model Driven Development of Multi-Agent Systems. In A. Rensink and J. Warmer, editors, *Model Driven Architecture Foundations and Applications*, volume 4066 of *Lecture Notes in Computer Science*, pages 284–298. Springer Berlin / Heidelberg, 2006.

[8] M. A. Wimmer, K. Furdik, M. Bicking, M. Mach, T. Sabol, and P. Butka. Open Collaboration in Policy Development: Concept and Architecture to integrate scenario development and formal policy modelling. In Y. Charalabidis and S. Koussouris, editors, *Empowering Open and Collaborative Governance*. Springer Berlin / Heidelberg, 2012.

[9] A. Winter. *Referenz-Meta-schema für visuelle Modellierungssprachen*. Deutscher Universitäts-Verlag, Oktober 2000.