SUMMARY

Background

Environmental computational models are abstract representations of natural systems that are used for decision support and to understand our complex environment. The domain scientists who develop these models possess domain knowledge, which implies that they have in mind which are the important objects and processes in their domain and how these are related. During the model development process they make choices about which aspects to include in their model, how to translate these into variables and computational relations, and how to implement these in software. In this way the domain knowledge of the scientists gets hidden in the models' source code. This domain knowledge is, however, essential to understand the meaning and context of the results and insights generated with these models. As a consequence, it is hard to make efficient and effective use of environmental computational models by other people than the original developers

The focus of this thesis is on computational models implemented as spreadsheets. Spreadsheets are widely used in the domain of natural science. The domain knowledge of the developers is implicitly presented in both the *content and structure* of the spreadsheet tables. The goal of our research is to explore to what extent we can make this domain knowledge behind natural science spreadsheets explicit. We model this knowledge at the "knowledge level", independent of the spreadsheet implementation and independent of the task, and refer to it as the *domain model*.

Approach

We investigate how this domain model can be adequately described, and how its reconstruction can be performed. Our research has an exploratory character and is based on the analysis of a limited number of case studies. Our case studies consist of spreadsheets, and other types of computational models from existing research projects in the domain of natural science.

We start our research by manually analyzing how the patterns in the text, structure and formulas of spreadsheet tables provide

insight in the semantics, and we record our insights in heuristics. We combine these heuristics with knowledge from external vocabularies, i.e., domain vocabularies and a vocabulary on quantities and units of measure, to automatically annotate the content of spreadsheet tables. These annotations make explicit which classes, instances and roles are implied by a spreadsheet table.

Furthermore, we map all computational dependencies between spreadsheet cells in a cell dependency graph. Subsequently, we combine the obtained heuristics with analysis of the formula syntax to semi-automatically aggregate all cells in the graph that represent instances and duplicates of the same quantities. The resulting set of interconnected quantities represents the calculation workflow, and provides insight in the way the classes, and instances in a table are computationally connected.

We evaluate the separate steps of the reconstruction of the domain model by comparing our automatically generated results with a manually constructed ground truth. Finally, we perform a survey on existing spreadsheets and theoretically analyze the plausibility of the reconstruction of the domain model in practice.

Results

We show that an adequate description of the domain model combines static domain knowledge, which comprises the classes, instances and properties, with dynamic inference knowledge, which comprises how the classes are connected through computational relationships.

The *structure* of spreadsheet tables contains implicit, but valuable information on the domain schema. Spreadsheet tables consist of rectangular blocks of semantically related cells which are characterized by their content and position, but also by their role in the table. We distinguish four types of blocks and define these with high-level concepts from the OM Ontology on quantities and units: 1) Measure, 2) Unit of Measure, 3) Quantity, and 4) Phenomenon. The phenomenon blocks are considered as classes in the domain schema, and the quantity instances as properties of these classes. The grouping of phenomenon cells determines which instances belong to the same class, and as such determines the hierarchical relationships. The cross-wise alignment of the phenomenon blocks and quantity instances reveals the property relationship between the two types. The quantities in a table serve as the connecting factors between the domain and the inference knowledge in the domain model

We present the sequential steps in the reconstruction process of the domain model, and show to what extent we can support these steps automatically. We observe that in practice the majority of the spreadsheets does not meet our requirements regarding ideal structure and content. By complementing our automatic reconstruction process with heuristics and knowledge from external vocabularies, we are able to reconstruct the domain model of existing natural science spreadsheets, even if the tables do not contain complete information.

Our final result is a set of guidelines on the design of natural science spreadsheets. These guidelines inform domain scientists how to make the knowledge in their spreadsheets explicit, without imposing too many constraints on the way they design their tables. The guidelines may also be implemented in spreadsheet software in order to provide automatic support during the development process.

Conclusions

We have demonstrated that modeling the knowledge encoded in environmental computational models provides a suitable way to make this knowledge explicit. The annotation of model code and data with concepts from external vocabularies adds meaning and context, so that these code and data can be used and understood by other people than the model developers. Moreover, these annotations turn isolated model code and data into linked data, which can be shared and reused in the Semantic Web.