# A Paradigm for Coupling Procedural and Conceptual Knowledge in Companion Systems

Marvin Schiller, Gregor Behnke, Mario Schmautz, Pascal Bercher, Matthias Kraus,
Wolfgang Minker, Birte Glimm, and Susanne Biundo
Faculty of Engineering, Computer Science and Psychology
Ulm University, Ulm, Germany

***Abstract*—We propose a modelling paradigm for integrating procedural and conceptual knowledge, which is targeted at companion systems that require a combination of planning and reasoning capabilities. The benefits of this approach are demonstrated with a prototype implementation.**

## I. Introduction

When developing a *companion system*[1] for being used in a complex technical domain, broadly two kinds of knowledge need to be incorporated into models of the application domain. Firstly, the set of available actions, how they contribute to the user's goals, and what constraints exist among them has to be known. They are the basis for the system to plan ahead and to present available options to the user. Secondly, the objects and concepts in the application domain, together with their semantic relationships, need to be represented in such a companion system. Both the fields of automated planning and semantic knowledge representation have brought forth their own dedicated formalisms. Hence, when both are used in a *companion system*, redundancies, or worse, inconsistencies between the models of the application domain might arise. Here, we propose a methodology that serves to tightly integrate a hierarchical planning formalism with ontology-based knowledge representation. It is based on the idea that the static relations between objects (i.e. the planner's state) are represented in an ontology in such a way that the actions of the planning model (describing the dynamics of the domain) directly incorporate the represented conceptualisations. Our paradigm strives to minimise any redundancies in modelling the planning domain and the modelling of conceptual knowledge. This facilitates coherence between the procedural and factual knowledge in the system's portfolio and helps to ease the issue of maintenance that arises when developing complex modularised systems. While the methodology itself is not limited to any particular application domain, we employ it in the context of a companion system which supports novice users with setting up and operating electrical equipment (e.g. a home theatre [1]).

## II. Preliminaries

We consider *hybrid planning* [2], [3], which combines hierarchical task network (HTN) planning [4], [5] with partial-order causal-link (POCL) planning [6], since it is well-suited for planning together with a user. Hybrid planning is a hierarchical planning approach that distinguishes between primitive tasks (i.e. directly executable actions) and abstract tasks, which are refined into more primitive courses of actions. Primitive tasks are defined by preconditions and effects.

As a knowledge-representation formalism, we consider an ontology formulated in OWL.[2] For the described approach, the restricted EL profile of OWL2 is sufficient, but more expressive languages could as well be used (e.g. OWL2 DL).

## III. Related Work

Gil [7] provides a survey of research that connects description logics and planning. More recent work is presented by Sirin [8]. Almost all such approaches use ontologies to define taxonomies on actions, plans, or goals. Our recent work [9] also falls in the line of action taxonomies. A fourth approach – defining taxonomies of objects – is quite similar to our proposed paradigm. However, all previously proposed planning formalisms use description logic expressions to describe their actions' preconditions and effects – instead of lists of literals. This poses a significant problem, as state-of-the-art planners are not able to handle these models and as it is not clear how heuristics can be extended to cover this formalism, as they are specifically designed for purely propositional planning formalisms. Our paradigm enables the use of state-of-the-art planners by clearly separating the planning model and the ontology.

## IV. A Paradigm for Coupling Procedural and Conceptual Knowledge

The guiding principle that we apply is that each model (the planning model representing procedural knowledge, the ontology representing factual/conceptual knowledge) manages only the information that is required to achieve its functionality, and to rely on the complementary model otherwise. The information that is shared between both models is kept lean to avoid unnecessary redundancy. Relationships between objects and categories are specified in the ontology. By generating the planner's problem description from it, we make it easily accessible to the planner, while retaining a standard semantics for the planner's actions. Accordingly, we design planning operators to only refer to those properties in the domain that

[1] By *companion systems* we refer to technical systems that adapt their functionality specifically to their user.

[2] Web Ontology Language, see https://www.w3.org/TR/owl2-profiles/

are relevant for the applicability of an action, such that the planning domain remains generic w.r.t. the individual objects that are available in a concrete planning problem.

Concepts are modeled as classes in the ontology and individual devices as instances thereof. Object property assertions model binary relationships between individuals in the domain, and data property assertions can be used to associate individuals with values such as character strings or numbers. For example, suppose that a particular device $d_1$ is an instance of a class of devices $D$. Suppose further that it is either specified in the ontology, or that it can be inferred from it that all devices $D$ are also devices of class $A$ (subsumption). Then $d_1$ is (by inference) also an instance of $A$.

The planning domain refers to the classes and properties specified in the ontology. Our approach translates class membership to a special binary predicate *hasType*, individuals to constants of type *Object*, and the class name to a constant of type *Type*. Thus, the preconditions of an action may specify, for example, that an object `?o` (e.g. a device) needs to be of type $A$ for an action to be applicable: `(hasType ?o A)` (using PDDL syntax [10]). In our example, the ontology component adds `(hasType d1 A)` to the initial state, among any other devices that are inferred to belong to the type $A$. Furthermore, binary relationships (object and data property assertions) in the ontology are made available to the planner in the initial state. For instance, if a device $d_1$ has a particular data property $p$ with a value $v$, the binary predicate `(p d1 v)` is added.

In complex technical domains, the applicability of an action sometimes depends on n-ary relationships (e.g. if an action requires a specific combination of device(s) and property values). Ontology languages, however, are typically restricted to binary relations (object property assertions and data property assertions). We reify such n-ary relationships by introducing an instance representing such a "configuration" in the ontology. Each individual parameter of the configuration is then assigned using binary object and data properties. Accordingly, the planning operators quantify over configurations and their binary predicates, instead of using n-ary predicates.

## V. Prototype Realization

A prototype system has been implemented that demonstrates the proposed methodology. Reasoning and planning are performed using the ontology reasoner JFact[3] and the planning system PANDA [11]. Thanks to the described approach, the planner did not – as usually in planning – need to be equipped with its own description of all the objects and their categories in the domain, but receives this information dynamically from the ontology component. Since all objects in the planning domain originate from the ontology, it is possible to query the ontology about their categories and relationships and additional information (e.g. descriptions and hints). An interface is provided to the system by a front-end web application created using the Angular4 framework.[4] It uses speech recognition (LUIS[5]) to identify planning goals. Generated plans are presented in the form of slides and synthesised speech.

## VI. Conclusion

Parametrising the planning domain by information represented in the ontology enables the integration of procedural and conceptual knowledge while maintaining a clean separation. As a result, the information in the ontology can be changed (e.g. devices added, properties added, etc.) and becomes dynamically available for planning. A further advantage of the tight integration of conceptual and procedural knowledge comes to bear when a companion system is required to deliver explanations for the plans it generates. To-be-explained relationships in the planning domain directly refer to concepts and relations in the ontology, such that both procedural and conceptual knowledge can be combined in the generated explanations.


## Acknowledgment

This work is done within within the technology transfer project "Do it yourself, but not alone: Companion-Technology for DIY support" of the Transregional Collaborative Research Centre SFB/TRR 62 "Companion-Technology for Cognitive Technical Systems" funded by the German Research Foundation (DFG).


## References


[1] P. Bercher, S. Biundo, T. Geier, T. Hörnle, F. Nothdurft, F. Richter, and B. Schattenberg, "Plan, repair, execute, explain - How planning helps to assemble your home theater," in *Proc. of ICAPS*, 2014, pp. 386–394.

[2] S. Biundo and B. Schattenberg, "From abstract crisis to concrete relief – a preliminary report on combining state abstraction and HTN planning," in *Proc. of ECP*, 2001, pp. 2–9.

[3] P. Bercher, D. Höller, G. Behnke, and S. Biundo, "More than a name? On implications of preconditions and effects of compound HTN planning tasks," in *Proc. of ECAI*, 2016, pp. 225–233.

[4] K. Erol, J. Hendler, and D. Nau, "Complexity results for HTN planning," *Annals of Mathematics and AI*, vol. 18, no. 1, pp. 69–93, 1996.

[5] T. Geier and P. Bercher, "On the decidability of HTN planning with task insertion," in *Proc. of IJCAI*, 2011, pp. 1955–1961.

[6] D. McAllester and D. Rosenblitt, "Systematic nonlinear planning," in *Proc. of AAAI*, 1991, pp. 634–639.

[7] Y. Gil, "Description logics and planning." *AI Magazine*, vol. 26, no. 2, pp. 73–84, 2005.

[8] E. Sirin, "Combining description logic reasoning with AI planning for composition of web services," Ph.D. dissertation, University of Maryland at College Park, 2006.

[9] G. Behnke, D. Ponomaryov, M. Schiller, P. Bercher, F. Nothdurft, B. Glimm, and S. Biundo, "Coherence across components in cognitive systems – One ontology to rule them all," in *Proc. of IJCAI*, 2015, pp. 1442–1449.

[10] M. Fox and D. Long, "PDDL2.1: An extension to PDDL for expressing temporal planning domains," *JAIR*, pp. 61–124, 2003.

[11] P. Bercher, S. Keen, and S. Biundo, "Hybrid planning heuristics based on task decomposition graphs," in *Proc. of SoCS*, 2014, pp. 35–43.


[3] http://jfact.sourceforge.net/

[4] https://angular.io/

[5] https://www.luis.ai/