

# The Epistemic Planning Domain Definition Language

Alessandro Burigana<sup>1</sup>, Francesco Fabiano<sup>2</sup>

<sup>1</sup>Free University of Bozen-Bolzano, Bolzano, Italy

<sup>2</sup>University of Parma, Parma, Italy

## Abstract

Epistemic planning is a branch of Artificial Intelligence that stems from the combination of automated planning and Dynamic Epistemic Logic (DEL). While DEL provides a very expressive framework—which comes at the cost of undecidability—to guarantee the feasibility of the planning process, various fragments have been studied in the literature, that rely on very specific syntax for representing domains. As a result, a comprehensive language that is able to capture the general DEL framework is currently not present in the literature. In this paper, we propose a new language called EPDDL (Epistemic Planning Domain Definition Language), through which we can capture the full DEL semantics, thus allowing for a general and unified syntax for representing epistemic planning domains.

## 1. Introduction

*Epistemic planning* [1] is an enrichment of automated planning, where the notions of *knowledge* and *belief* [2] are introduced. In this field, we take into account the perspective of one or more agents: *epistemic states* contain both factual information about the world and epistemic information concerning the knowledge/beliefs of agents. Similarly, *epistemic actions* describe how their knowledge/beliefs change. For instance, imagine that Anne received a letter from an university she applied for. She then starts reading it in front of Bob, who can not directly take a look at it. Once Anne has finished reading, her perspective will differ from Bob's: while she now *knows* whether she was admitted or not, Bob still considers both options to be possible. Thus, the epistemic action describing this situation has to consider the points of view of both agents.

Epistemic planning has been frequently built on the formalism of Dynamic Epistemic Logic (DEL) [3]. Since the general framework is undecidable, various epistemic planners implement fragments of DEL [4, 5, 6, 7, 8, 9]. One of the main issues is that, to represent planning domains, either solvers rely on languages tailored for a specific fragment [6, 10, 11], or they adopt no language at all, making the comparison of existing solvers more difficult.

In this paper, we introduce the *Epistemic Planning Domain Definition Language* (EPDDL). The language stems from the combination of the syntax of PDDL and the DEL semantics and it aims at overcoming the specificity of existing languages. Namely, EPDDL is able to handle the full DEL framework, which in turn supports an unified representation. The paper is organized as follows. In Section 2, we recall the semantics of DEL. In Section 3, we describe EPDDL and its main features. We conclude with a brief discussion and some future works.

---

IPS 2022: 10th Italian Workshop on Planning and Scheduling, November 28–December 2, 2022, Udine, Italy

✉ burigana@inf.unibz.it (A. Burigana); francesco.fabiano@unipr.it (F. Fabiano)

🆔 0000-0002-9977-6735 (A. Burigana); 0000-0002-1161-0336 (F. Fabiano)



© 2022 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

## 2. Dynamic Epistemic Logic

**Epistemic Models.** Let  $\mathcal{P}$  be a countable set of propositional atoms and  $\mathcal{AG} = \{1, \dots, n\}$  be a finite set of agents. The language  $\mathcal{L}_{\mathcal{P}, \mathcal{AG}}^C$  of *multi-agent epistemic logic* is defined by the BNF:  $\varphi ::= p \mid \neg\varphi \mid \varphi \wedge \psi \mid \Box_i\varphi \mid C_G\varphi$ , where  $p \in \mathcal{P}$ ,  $i \in \mathcal{AG}$  and  $G \subseteq \mathcal{AG}$ . We read the formulae  $\Box_i\varphi$  and  $C_G\varphi$  as “agent  $i$  knows/believes that  $\varphi$ ” and “group  $G$  has common knowledge/belief that  $\varphi$ ”, respectively. In the *possible world* semantics of DEL, a *Kripke model* [12] represents an *epistemic model*. Epistemic models contain both factual information on multiple possible worlds and epistemic information, *i.e.*, which worlds are considered possible by agents.

**Definition 1 (Kripke Model).** A Kripke model is a triple  $M = (W, R, V)$  where: 1)  $W \neq \emptyset$  is the set of possible worlds, 2)  $R : \mathcal{AG} \rightarrow 2^{W \times W}$  assigns to each agent  $i$  an accessibility relation  $R(i)$  (abbreviated as  $R_i$ ), 3)  $V : \mathcal{P} \rightarrow 2^W$  assigns to each atom a set of worlds.

An *epistemic state* is a pair  $(M, W_d)$ , where  $W_d \subseteq W$  is a non-empty set of *designated worlds* denoting the *real state* of affairs. Truth of formulae is defined as follows: i.  $(M, w) \models p$  iff  $w \in V(p)$ ; ii.  $(M, w) \models \neg\varphi$  iff  $(M, w) \not\models \varphi$ ; iii.  $(M, w) \models \varphi \wedge \psi$  iff  $(M, w) \models \varphi$  and  $(M, w) \models \psi$ ; iv.  $(M, w) \models \Box_i\varphi$  iff  $\forall v$  if  $(w, v) \in R_i$  then  $(M, v) \models \varphi$ ; and v.  $(M, w) \models C_G\varphi$  iff  $\forall v$  if  $(w, v) \in R_G^*$  then  $(M, v) \models \varphi$  (where  $R_G = \cup_{i \in G} R_i$  and  $R_G^*$  denotes its transitive closure). Finally,  $(M, W_d) \models \varphi$  iff  $(M, v) \models \varphi$ , for all  $v \in W_d$ .

**Event Models.** In DEL, actions are modeled by *event models* [13, 14], that represent how new information changes the perspectives of agents. Here, *events* can be seen as *possible actions* and the accessibility relation  $Q_i$  describes which events are considered possible by agent  $i$ .

**Definition 2 (Event Model).** An event model is a quadruple  $\mathcal{E} = (E, Q, pre, post)$  where: 1)  $E \neq \emptyset$  is the set of events, 2)  $Q : \mathcal{AG} \rightarrow 2^{E \times E}$  assigns to each agent  $i$  an accessibility relation  $Q(i)$  (abbreviated as  $Q_i$ ), 3)  $pre : E \rightarrow \mathcal{L}_{\mathcal{P}, \mathcal{AG}}^C$  assigns to each event a precondition, 4)  $post : E \rightarrow (\mathcal{P} \rightarrow \mathcal{L}_{\mathcal{P}, \mathcal{AG}}^C)$  assigns to each event a postcondition for each atom.

Informally, preconditions capture the applicability of events, whereas postconditions determine whether an atom is true after the event is applied. A *(multi-)pointed event model* is a pair  $(\mathcal{E}, E_d)$ , where  $E_d \subseteq E$  is a non-empty set of *designated events*, and it represents an *action* in DEL.

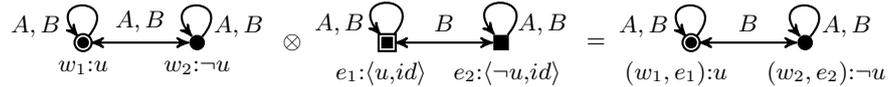
**Product Update.** The product update formalizes how actions bring about information change in epistemic states in DEL. An action  $(\mathcal{E}, E_d)$  is *applicable* in  $(M, W_d)$  if and only if for each world  $w \in W_d$  there exists an event  $e \in E_d$  such that  $(M, w) \models pre(e)$ .

**Definition 3 (Product Update).** Given an action  $(\mathcal{E}, E_d)$  applicable in an epistemic state  $(M, W_d)$ , where  $M = (W, R, V)$  and  $\mathcal{E} = (E, Q, pre, post)$ , the product update of  $(M, W_d)$  with  $(\mathcal{E}, E_d)$  is the multi-pointed Kripke model  $(M, W_d) \otimes (\mathcal{E}, E_d) = ((W', R', V'), W'_d)$ , where:

1.  $W' = \{(w, e) \in W \times E \mid (M, w) \models pre(e)\}$ ,
2.  $R'_i = \{((w, e), (v, f)) \in W' \times W' \mid (w, v) \in R_i \text{ and } (e, f) \in Q_i\}$ ,
3.  $V'(p) = \{(w, e) \in W' \mid (M, w) \models post(e)(p)\}$ ,
4.  $W'_d = \{(w, e) \in W' \mid w \in W_d \text{ and } e \in E_d\}$ .

**Example 1 (DEL).** Consider the above situation with Anne (A) and Bob (B). Before Anne reads the letter, both agents consider possible two options: either she was admitted in the university

( $u$ ), or not. Thus, we need two worlds to represent this uncertainty ( $w_1$  and  $w_2$ , respectively). As we are universal observers, we assume that we know that  $w_1$  is the designated world (circled dot). This is modeled by the epistemic state on the left. When Anne reads the letter, Bob can not read its content. In the center of the picture, events  $e_1$  (Anne learns that  $u$ ) and  $e_2$  (Anne learns that  $\neg u$ ) are needed to represent Bob's uncertainty about the outcomes of the action (we use the notation  $\langle pre, post \rangle$  for events). In this way, Anne learns that she was admitted, while Bob remains uncertain. Moreover, both Anne and Bob are reciprocally aware of their perspective. The epistemic state  $M'$  that results from this action is on the right. Notice that now Anne knows that she was admitted, i.e.,  $(M', (w_1, e_1)) \models \Box_a u$ , while Bob does not change his perspective.



### 3. EPDDL

The development of the syntax of EPDDL was driven by three design principles: **(1)** to maintain the style of PDDL syntax, **(2)** to capture the *entire* DEL semantics (i.e., to be able to represent *all* possible event models), and **(3)** to obtain an intuitive and usable language, even for researchers that are less familiar with epistemic planning and DEL. The main structure of EPDDL includes three main components: *problem* (specific part), *domain* and *action type library* (universal parts). Due to space limits, we do not provide the full BNF and we base our exposition on Example 1.

**Problem.** The problem defines “specific” aspects of a planning task. In EPDDL, in addition to objects, initial state and goal, we also list which *agents* are present in the particular instance. Goals in EPDDL are expressed by a generic formula  $\varphi_g \in \mathcal{L}_{P,AG}^C$ . The syntax of propositional formulae is the same as in PDDL. Modal formulae such as  $\Box_i \varphi$  and  $C_G \varphi$  (with  $|G| \geq 2$ ) are represented as  $[i] \varphi$  and  $[G] \varphi$ , respectively. Finally, initial states can be represented in two ways: either explicitly (specifying worlds, relations and valuation), or by means of a *finitary S5-theory* [15], i.e., a *set of formulae* of a particular form that admits a finite number of (equivalent) finite models (computable in polynomial time). In our example, we use a finitary S5-theory to state that: 1) Anne was admitted to the university (line 4), and 2) Anne and Bob have common knowledge that they both do not know whether she was admitted (lines 5-6).

```

1 (define (problem p1)           4 (:init (u))
2   (:domain example1)         5   [Anne Bob] (and (not [Anne](u)) (not [Anne] (not (u))))
3   (:agents Anne Bob)         6   [Anne Bob] (and (not [Bob](u)) (not [Bob] (not (u))))
                                7   (:goal [Anne](u))

```

**Domain and Action Type Library.** In EPDDL, the universal aspects of a problem (types, predicates, actions) are jointly described by a *domain* and an *action type library*. Moreover, actions are defined on three separated levels of abstraction: *events*, *action types* and *actions*. Domains define types, predicates and actions, whereas action type libraries contain the description of events and action types. We now analyze more in depth these elements. Events constitute the atomic components, where preconditions and postconditions (if any) are defined. Events are then combined within action types to represent event models (Definition 2). Finally, each action must include its *type* in its specification. Actions and their type are described separately, since

it is common for different actions to share the same underlying structure. This allows for a more concise and readable representation (design principle (3)).

Since a library is intended to describe *universal* aspects, action types should not refer to specific entities of a problem. To this end, we implemented two important design choices: 1) parametrized events and action types, and 2) *observability groups*. First, parameters allow to abstract from particular predicates and agents. Importantly, in EPDDL, apart from objects, parameters can also refer to *agents*, *formulae* and *postconditions*. This aspect is crucial to maintain the *universality* of action types: for instance, if we pass the preconditions of events as parameters, we can simply refer to them as variables within an action type. As a result, action type libraries can be used transversally across *different domains*. Second, observability groups generalize accessibility relations. Each group represents the perspective of one or more agents. For instance, when Anne reads the letter, she is *fully observant*, since she learns its content, while Bob is *partially observant*, since he remains uncertain about Anne’s knowledge. Action types only refer to observability groups (lines 13-15). Then, in each action we assign each agent to a group (lines 29-33). We now show the EPDDL representation of the action of Example 1.

```

1 (define (library lib)
2   (:event e1
3     :parameters (?sensed - predicate)
4     :precondition (?sensed))
5   (:event e2
6     :parameters (?sensed - predicate)
7     :precondition (not (?sensed)))
8   (:action-type sensing
9     :parameters (?p - predicate)
10    :observability-groups (Fully Partially)
11    :events (e1 (?sensed :: ?p) )
12            (e2 (?sensed :: ?p) )
13    :relations (Fully (e1 e1) (e2 e2))
14              (Partially (e1 e1) (e2 e2)
15                        (e1 e2) (e2 e1))
16    :designated (e1)))
17 (define (domain example1)
18   (:action-type-libraries lib)
19   (:requirements :del :typing :equality
20                 :universal-conditions)
21   (:predicates (u)
22                (has_letter ?ag - agent))
23   (:action read_letter
24     :parameters (?ag - agent)
25     :action-type (sensing (?p :: (u)) )
26     :precondition (has_letter ?ag)
27     :observability-conditions
28       (?ag Fully)
29     (forall (?ag2 - agent)
30       (if (not (= ?ag2 ?ag))
31           (Partially)
32           ))))

```

## 4. Discussion

In this paper, we have introduced a new language for describing epistemic planning domains, called EPDDL. Importantly, this language is able to capture the full DEL semantics. Intuitively, this follows from the fact that each component of an event model (events, relations, preconditions, postconditions) is captured by a corresponding syntactical element of EPDDL. Thus, we obtain a syntax through which we can represent the existing fragments of epistemic planning in an unified way. Due to space limits, we could not address all features of the language. As future work, we plan on finalizing the syntax of EPDDL and to implement a full-fledged parser (also including a *type checker*). Moreover, we intend to use EPDDL to create a public repository of epistemic planning domains to be used as benchmarks for epistemic planners.

## References

- [1] T. Bolander, M. B. Andersen, Epistemic Planning for Single and Multi-Agent Systems, *J. Appl. Non Class. Logics* 21 (2011) 9–34. URL: <https://doi.org/10.3166/jancl.21.9-34>. doi:10.3166/jancl.21.9-34.
- [2] J. Hintikka, *Knowledge and Belief: An Introduction to the Logic of the Two Notions*, Contemporary Philosophy, Cornell University Press, 1962.
- [3] H. P. van Ditmarsch, W. van der Hoek, B. P. Kooi, *Dynamic Epistemic Logic*, volume 337, Springer Netherlands, 2007. URL: <https://doi.org/10.1007/978-1-4020-5839-4>. doi:10.1007/978-1-4020-5839-4.
- [4] A. Burigana, F. Fabiano, A. Dovier, E. Pontelli, Modelling Multi-Agent Epistemic Planning in ASP, *Theory Pract. Log. Program.* 20 (2020) 593–608. URL: <https://doi.org/10.1017/S1471068420000289>. doi:10.1017/S1471068420000289.
- [5] F. Fabiano, A. Burigana, A. Dovier, E. Pontelli, EFP 2.0: A Multi-Agent Epistemic Solver with Multiple E-State Representations, in: J. C. Beck, O. Buffet, J. Hoffmann, E. Karpas, S. Sohrabi (Eds.), *Proceedings of the Thirtieth International Conference on Automated Planning and Scheduling*, Nancy, France, October 26-30, 2020, AAAI Press, 2020, pp. 101–109. URL: <https://aaai.org/ojs/index.php/ICAPS/article/view/6650>.
- [6] X. Huang, B. Fang, H. Wan, Y. Liu, A general multi-agent epistemic planner based on higher-order belief change, in: C. Sierra (Ed.), *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017*, Melbourne, Australia, August 19-25, 2017, [ijcai.org](http://ijcai.org), 2017, pp. 1093–1101. URL: <https://doi.org/10.24963/ijcai.2017/152>. doi:10.24963/ijcai.2017/152.
- [7] F. Kominis, H. Geffner, Beliefs In Multiagent Planning: From One Agent to Many, in: R. I. Brafman, C. Domshlak, P. Haslum, S. Zilberstein (Eds.), *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling, ICAPS 2015*, Jerusalem, Israel, June 7-11, 2015, AAAI Press, 2015, pp. 147–155. URL: <http://www.aaai.org/ocs/index.php/ICAPS/ICAPS15/paper/view/10617>.
- [8] T. Le, F. Fabiano, T. C. Son, E. Pontelli, EFP and PG-EFP: Epistemic Forward Search Planners in Multi-Agent Domains, in: M. de Weerdt, S. Koenig, G. Röger, M. T. J. Spaan (Eds.), *Proceedings of the Twenty-Eighth International Conference on Automated Planning and Scheduling, ICAPS 2018*, Delft, The Netherlands, June 24-29, 2018, AAAI Press, 2018, pp. 161–170.
- [9] C. J. Muise, V. Belle, P. Felli, S. A. McIlraith, T. Miller, A. R. Pearce, L. Sonenberg, Planning Over Multi-Agent Epistemic States: A Classical Planning Approach, in: B. Bonet, S. Koenig (Eds.), *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, January 25-30, 2015, Austin, Texas, USA, AAAI Press, 2015, pp. 3327–3334. URL: <http://www.aaai.org/ocs/index.php/AAAI/AAAI15/paper/view/9974>.
- [10] C. Baral, G. Gelfond, E. Pontelli, T. C. Son, An Action Language for Multi-Agent Domains: Foundations, *CoRR* abs/1511.01960 (2015). URL: <http://arxiv.org/abs/1511.01960>.
- [11] F. Fabiano, B. Srivastava, J. Lenchner, L. Horesh, F. Rossi, M. B. Ganapini, E-PDDL: A standardized way of defining epistemic planning problems, *CoRR* abs/2107.08739 (2021). URL: <https://arxiv.org/abs/2107.08739>. arXiv:2107.08739.
- [12] S. A. Kripke, Semantical Considerations on Modal Logic, *Acta Philosophica Fennica* 16

(1963) 83–94.

- [13] A. Baltag, L. S. Moss, S. Solecki, The Logic of Public Announcements and Common Knowledge and Private Suspicions, in: I. Gilboa (Ed.), Proceedings of the 7th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-98), Evanston, IL, USA, July 22-24, 1998, Morgan Kaufmann, 1998, pp. 43–56.
- [14] H. van Ditmarsch, B. Kooi, Semantic Results for Ontic and Epistemic Change, Texts in Logic and Games 3, Amsterdam University Press, 2008, pp. 87–117.
- [15] T. C. Son, E. Pontelli, C. Baral, G. Gelfond, Finitary s5-theories, in: E. Fermé, J. Leite (Eds.), Logics in Artificial Intelligence - 14th European Conference, JELIA 2014, Funchal, Madeira, Portugal, September 24-26, 2014. Proceedings, volume 8761 of *Lecture Notes in Computer Science*, Springer, 2014, pp. 239–252. URL: [https://doi.org/10.1007/978-3-319-11558-0\\_17](https://doi.org/10.1007/978-3-319-11558-0_17). doi:10.1007/978-3-319-11558-0\_17.