# Plan execution in a temporal logic environment *

**D. Dengler**

German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany
phone: +49 (681) 302-5259
fax: +49 (681) 302-5341
e-mail: dengler@dfki.uni-sb.de

## Introduction

This extended abstract describes essential aspects of a current application scenario which was initiated in order to investigate the applicability of research results of the RAP project in a rather real-world environment. The work on this scenario is not completed which means that reported results are rather preliminary.

The project *RAP* (Reasoning About Plans) (Bauer *et al.* 1996) aims at a logic-based shell which supports the consistent modeling of planning knowledge and provides a generic module for the logic-based reasoning about plans which can be integrated into a large variety of application systems. The RAP shell, for example, enables users to graphically configure their own complex reasoning procedures and control systems based on basic reasoning services like *plan generation*, *plan validation* and *verification*, *temporal projection*, *plan optimization*, *plan modification*, *symbolic plan execution*, *plan interpretation*, etc.

Our current application scenario is about an autonomous robot indoor simulation environment in an extended logistic setting. There is a building consisting of a shop with several devices for the analysis and manipulation of objects, and there are differently classified storage rooms according to different properties objects can have. The modeled robot tasks include movements, simple processing, storage, and disposal of certain objects.

In order to control the activities of the robot we have built a plan execution control system as a special plan reasoning system in the RAP shell. The graphical simulation environment has been integrated as a front-end to the control system. Figure 1

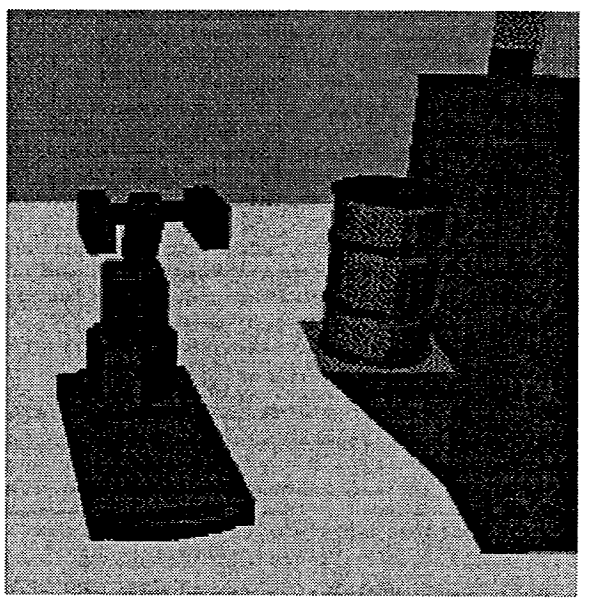

Figure 1: Sample view on the robot simulation

*This work was supported by the German Ministry for Education, Science, Research and Technology (BMBF) under contract ITW 9404 0 as part of the RAP project.

*The reported work partially describes results originating from a cooperative work with my colleagues M. Bauer and especially H. Feibel.

shows e.g. our robot putting a container in some device. The dynamism in our domain is characterized by the possible occurrence of changes in the world state which are not caused by activities of our robot.

In the following we first introduce our notion of plans and the characteristics which facilitate their execution in our dynamic robot environment. Then, we present the above mentioned plan execution model in more detail.

## The nature of executable plans

As a common basis for all plan inference procedures integrated in the RAP system we have built a compositional temporal logic framework. Planning, for example, is viewed as a logical inference process based on various refinements and plans work as transmitters of appropriate information about this process. The temporal logic enables us to reach representational and operational flexibility in order to handle a wide variety of behavioral constraints and plans of arbitrary abstraction levels. The proposed framework is built upon a well-known temporal logic version (Manna & Pnueli 1981; Rosner & Pnueli 1986) successfully used for the specification and verification of concurrent processes and programs (Manna & Pnueli 1992). We slightly adapted the setting to our needs for planning purposes (Biundo, Dengler, & Koehler 1992). Owing to lack of space we concentrate only on the aspects of the logic necessary to understand the topics discussed here.

The logic is a first-order modal logic whose models are based on intervals each consisting of a sequence of states. For example, a finite interval $\sigma = \langle \sigma_0 \ldots \sigma_n \rangle$ consists of $n+1$ states, where $0 \leq n$. A state $\sigma_i$ is of two parts: $\sigma_i^1$ is an interpretation of so-called *local variables* which can change their values over time; $\sigma_i^2$ is a mapping to an action term $a_i$ stating that $a_i$ designates the action executed in state $\sigma_i$. On the object level there is a special predicate $ex$ which takes a term $a_i$ as an argument. A formula $ex(a_i)$ is interpreted as *true* at a state $\sigma_i$ *iff* $\sigma_i^2$ maps to $a_i$, so it designates the execution of an action. We denote the sequence $\sigma^A = \langle \sigma_0^2 \ldots \sigma_{n-1}^2 \rangle$ associated with $\sigma$ as the *action interval* w.r.t. $\sigma$. The corresponding sequence $\sigma^D = \langle \sigma_0^1 \ldots \sigma_n^1 \rangle$ is denoted as the *description interval* w.r.t. $\sigma$.

The interpretation of all logical variables, functions and other predicates is static. As a consequence, in order to express that propositions change their truth values over time they must take a local variable as an argument. An appropriate method to represent a state of the world is to view the dynamic entities of the world as complex objects bound to local variables. The objects have a lot of specific properties whose values can be changed by actions. An object is characterized by a set of equations of the kind "*object_property = value*".[1] In the following, if we talk about propositions, preconditions, effects, or goals then, if not otherwise stated, the atomic formulae concerning entities of the world are equations of the kind just mentioned.[2]

We will now informally introduce the semantics of the temporal modal operators we use. As a prerequisite, it is necessary to know, that all formulas in our logic are interpreted over intervals.
Let $\sigma = \langle \sigma_0 \sigma_1 \ldots \rangle$ be a possibly infinite interval.

- The formula $\phi$, where $\phi$ does not contain any modal operator, holds in the interval $\sigma$ if $\phi$ holds in $\sigma_0$, i.e. the interpretation of a modal-free formula only depends on the current state. An *ex*-predicate is interpreted w.r.t. $\sigma_0^2$ and other parts of $\phi$ are interpreted in $\sigma_0^1$.

- The formula $\circ\phi$ (*next* $\phi$) holds in $\sigma$ if $\phi$ holds in the terminal interval $\sigma' = \langle \sigma_1 \sigma_2 \ldots \rangle$, i.e. we have a lookahead of a single state transition.[3]

- The formula $\Diamond\phi$ (*sometimes* $\phi$) holds in $\sigma$ if there exists a $n$, $0 \leq n$ such that $\phi$ holds in the interval $\sigma' = \langle \sigma_n \sigma_{n+1} \ldots \rangle$, i.e. $\phi$ depends on some terminal subinterval or on $\sigma$ itself.

- The formula $\Box\phi$ (*always* $\phi$) holds in $\sigma$ if $\phi$ holds in every terminal subinterval of $\sigma$ including $\sigma$ itself.

- The formula $\phi;\psi$ ($\phi$ *chop* $\psi$) holds in $\sigma$ if $\phi$ holds in an initial finite subinterval $\sigma' = \langle \sigma_0 \sigma_1 \ldots \sigma_n \rangle$ and $\psi$ holds in the terminal subinterval $\sigma'' = \langle \sigma_n \sigma_{n+1} \ldots \rangle$, i.e. the operator *chop* realizes the temporal composition of two formulae.

Actions occuring in the planning domain are represented in our framework by axioms of the following structure:

$$\forall x \; pre(x) \wedge ex(a(x)) \rightarrow \circ eff(x)$$

Therein, $pre(x)$ is a modal-free formula describing the preconditions for action $a(x)$ and $eff(x)$ a modal-free formula representing the effects of $a(x)$,

[1] The time-dependent interpretation of predicates which is often used in other planning languages can be encoded by the use of boolean-valued functions.

[2] Additionally, codesignation and noncodesignation constraints can exist.

[3] As a rather technical note: We use *weak next* semantics in order to appropriately handle finite interval s.

respectively.[4] The interpretation in our temporal framework is as follows: if $pre(x)$ and $ex(a(x))$ hold in a certain state then it can be deduced that $eff(x)$ holds in the immediately accessible next state; we then also say that the action $a(x)$ is executed. That is the usual view on actions as a state-changing function describing the changes in the world caused by the actions. The action axioms are derived in a *call-by-need* fashion from more general axiom schemata according to well-known regression as well as progression mechanisms. It is e.g. possible to get axiom instances for an action $a(x)$ describing the weakest preconditions for a given formula w.r.t. $a(x)$ as well as describing the strongest postconditions of a formula w.r.t. $a(x)$. The axiom schema of an action specifies which properties of objects are assigned changed values by the execution of the action; only the minimal changes of the action affecting the world are specified. Now, knowing what actions are, we introduce the notion of a plan specification in order to be able to describe a planning problem. We think of plan specifications as schemata of the following structure:

$$\forall x\, precondition(x) \wedge \mathsf{Plan} \rightarrow goal(x)$$

Therein, $precondition(x)$ and $goal(x)$ are temporal logic formulae and Plan is a metavariable (a placeholder) for a formula which is currently unknown. In most cases, $precondition(x)$ is a modal-free formula specifying the initial state of the planning problem, $goal(x)$, however, is in general an arbitrary modal formula specifying the goals which has to be reached. The role of a plan specification is to specify the temporal behavior of a plan which satisfies the specification formula. During the planning process an instantiation $P$ for the metavariable Plan has to be found such that a model exists w.r.t. the action axioms specified which satisfies the following requirements: if $precondition(x)$ and $P$ hold in that model then $goal(x)$ does also hold. There is a restriction on the models we are interested in. They have to satisfy the following constraint: Let $\sigma = \langle \sigma_0 \sigma_1 \ldots \rangle$ be the interval associated with a model, for each $\sigma_i^2$ with value $a_i(y)$ there must exist an action axiom $pre(y) \wedge ex(a_i(y)) \rightarrow eff(y)$ such that $pre(y)$ holds in $\sigma_i^1$. We will denote such models as *action models*. Given a plan specification, there are in general a lot of action models which can serve as models for the specification formula. A plan—the instantiation of the metavariable Plan—is in our sense a characterization of these models in terms of a temporal logic formula. The minimal requirement on an instantiation of Plan is that it is at least as specific as the formula $goal(x)$ is. Since the plan is casted in the role of a transmitter of information arisen and available during planning there is no new information content (besides the action model existence) if Plan is instantiated by $goal(x)$; so, it is usually instantiated by a more specific formula. Planning is performed in our approach by the conduction of proofs in a sequent calculus using tactical theorem proving techniques (Dengler 1994). We should summarize the main characteristics of our planning perspective discussed so far:

- plan specifications (planning problems) and plans are represented in the same temporal logic language providing flexibility and semantics for free, full-fledged reasoning about plans can be done;
- planning means to guarantee the existence of at least one action model for a given plan specification;
- a plan is a characterization of appropriate action models and works as a transmitter of information produced during planning.

Details about the flexible planning process can be found in (Dengler 1996).

Now, how can this framework be used to handle plan execution in our robot simulation domain. As a prerequisite we first discuss the representation of the domain:

- A world state $S$ is described by a formula $\bigwedge_{i=1}^{n} \bigvee_{j=1}^{m_i} gaf_{ij}$ where each $gaf_{ij}$ is a ground atomic formula; the disjunctions can be used to express the uncertainty about concrete values of some properties or about the satisfaction of certain facts.
- Actions the robot can perform are defined by a set $A$ of action axiom schemata; currently, only deterministic actions are available.
- Other events, which are able to change a world state, are also given in the form of action axiom schemata by a set $E$; a simultaneous occurrence of actions and events is not allowed.
- There is a set $SC$ of constraints which define certain rules of behavior for the robot. They are given as modal-free formulae defining properties of world states which are not allowed to be disturbed or must be maintained, respectively. For example, it is possible to define a "safety condition" demanding that whenever two objects of a certain type are

[4] $pre(x)$ and $eff(x)$ are also parameterized by local variables, but we omit these details here.

inside the same storage room, then they must be put on a pallet.

- The task the robot must perform is defined by the specification of the goal which must be reached; this goal $G$ is given by a temporal logic formula, e.g. by a formula $\Diamond \bigwedge_{i=1}^{k} gaf_i$ specifying that a set of specific propositions should be satisfied at a certain point in the future.

- A plan $P$ describes the behavior of the robot or the way things in the world change in order to reach the goal.

We will now discuss properties of a useful plan to control a robot in the dynamic environment.

- The plan should be partial or incremental
  Given the robots current task, it makes no sense to develop a fully-detailed complete plan in advance, since a lot of things can happen in the world which cannot be foreseen during plan construction time. An appropriate plan specifies only a few steps of the robot's activity in detail and the end of the plan is for the present given as a more or less structured goal description, i.e. for a small interval of time the robot has concrete instructions what it should do, after that interval is passed it is only specified what it should reach.

  For example, given a world state $S$ and a robot task specified by $\Diamond g_1$ an incremental plan could result as:

  $$ex(a_1) \wedge skip\ ;\ ex(a_2) \wedge skip\ ;\ \Diamond g_0\ ;\ \Diamond g_1$$ [5]

  The robot should perform a sequence of two actions $a_1$ and $a_2$ in state $S$, then it should sometimes reach the subgoal $g_0$ in order to reach finally $g_1$. Considering the semantics of our logical framework the given plan is a compact description of a set of action models w.r.t. the goal $\Diamond g_1$ all with a common fixed action prefix. As time passes the plan can be further refined considering the corresponding world state.

  The quantitative relation between concrete instructions and structured goals depends upon the degree of dynamism in the domain and the complexity of the current task. An example in the robot simulation is e.g. the task of transportation of object $o$ from storage room $a$ to a shelf in storage room $b$. The plan is concrete to the point where the robot leaves room $a$; then a structured goal specifies that room $b$ must be entered and after that $o$ must be sometimes in the appropriate shelf.

- The plan should consider benevolent changes in the world and action variability
  Although an incremental plan only specifies a few concrete actions it nevertheless demands that they are executed. But, before these actions are scheduled for execution the world could have been benevolently changed in a way that some of these actions are now unnecessary to execute, or if they should still be executed it could be necessary to first introduce a particular repair plan in order to reconstruct their necessary preconditions. But essentially, the only motivation for proposing the actions was to reach a world state with specific properties.

  We realize that view w.r.t. the incremental plan given above as an abstracted plan:[6]

  $$\begin{array}{l} ends(\mathit{eff}_{a_1}) \wedge maxskip\ ; \\ ends(\mathit{eff}_{a_2}) \wedge maxskip\ ; \\ \Diamond g_0\ ;\ \Diamond g_1 \end{array}$$

  This plan starts with a dense sequence of two goals $\mathit{eff}_{a_1}$ and $\mathit{eff}_{a_2}$ associated with the actions $a_1$ and $a_2$, respectively, and ends again with the structured more abstract goals. What are the benefits? The plan delays the decision whether actions $a_1$ or $a_2$ are really necessary to execution time. It only describes that the satisfaction of the corresponding subgoals is useful in order to fulfill the complete task.

  Another point is that there is the option at runtime to possibly choose one from a set of actions which all reach $\mathit{eff}_{a_1}$ or $\mathit{eff}_{a_2}$, respectively; perhaps, action $a_1$ chosen during plan construction time is no more applicable.

- The plan should only contain local causal links
  This is a plan construction heuristic and has no effect on the syntactic form of a plan. It directs the choice between alternatives in goal/subgoal and goal/action refinements in a way that the distance (measured in state transitions) between the producer and consumer of a proposition should not exceed a fixed limit. The motivation is that the persistence of a proposition is rather time-limited in a dynamic domain.

[5] $skip$ is an abbreviation for a temporal formula $\neg\circ\neg\circ false$ constraining an interval to have exactly length 1.

[6] $ends(\phi)$ is an abbreviation for the formula $\Diamond[\phi \wedge \circ false]$ constraining an interval such that $\phi$ holds in its last state. $maxskip$ is an abbreviation for a temporal logic formula restricting an interval to have at most length 1.

- The plan can secure its causal links
  If a plan has causal links over a distance of more than one state transition, i.e. there is an interval preservation constraint on an interval with a length greater than 1, then this link can be partially or fully made explicit in the plan.

  A sample plan containing a general causal link with condition $\phi$ is given as:

$$ex(a_1) \wedge skip\ ;\ \left[\begin{array}{c} \Box\phi \wedge \\ ends(eff_2) \wedge maxskip\ ;\ ends(eff_3) \wedge maxskip \end{array}\right];\ \Diamond g_0\ ;\ \Diamond g_1$$

  This plan represents that action $a_1$ produces a condition $\phi$ which is needed as a precondition for the not yet constructed subplan reaching the subgoal $g_0$. Furthermore, the preservation condition demands that the actions, which are used to refine the immediate goals $eff_2$ and $eff_3$ are constrained not to violate the condition $\phi$—a local constraint on the robot's activity. Considering events which can unpredictably change the world a runtime check of the satisfaction of $\phi$ allows in the case of a violation a premature adaptation of the robot's strategy performing its task—a hint on a rather global redirection of the robot's activity.

## A plan execution system

Now, knowing the essentials of our temporal logic framework and important aspects of executable plans which can be defined in it, we introduce a plan execution system which deals with these plans. The execution system consists of four interrelated processes:

- A deliberative planner
  It gets as input the current world state $S$, the action axioms $A$, behavior constraints $SC$, and a task as a goal description $G$; as output it will deliver an incremental plan $P$ with the special properties described above. Firstly, an appropriate planning problem is constructed as a plan specification

$$pre \wedge \mathsf{Plan} \rightarrow G, \quad \text{where } S \rightarrow pre \text{ holds}$$

  The condition $pre$ is a result of some filtering applied on state $S$, e.g. to restrict the current domain knowledge available to the deliberation process. An appropriately directed refinement process, working on the plan specification, ends in an instantiation $P$ for the placeholder Plan. The plan is one of the inputs of the subsequent plan interpretation process.

- A plan interpreter
  The other inputs are the current state $S$ and action axioms $A$. The task of the plan interpreter is to refine the following problem in a particular way:

$$S \wedge [\mathsf{Step} \wedge maxskip]\ ;\ P' \rightarrow P$$

  Here, it is specified that we are looking for the next action (an instantiation for the placeholder Step) the robot should perform in order to follow plan $P$. The formula $P'$ represents the rest of the plan. The following tabular lists explain by abstract examples for a given $P$ possible solutions to the problem above:

$P$

| $ex(a_1) \wedge skip\ ;\ ex(a_2) \wedge skip\ ;\ \Diamond g_1$ | |
|---|---|
| Step | $P'$ |
| $ex(a_1)$ | $ex(a_2) \wedge skip\ ;\ \Diamond g_1$ |

$P$

| $ends(eff_1) \wedge maxskip\ ;\ \Diamond g_1$ | |
|---|---|
| Step | $P'$ |
| $ex(a_1)$ | $\Diamond g_1$ |

$P$

| $ends(eff_1) \wedge maxskip\ ;\ ends(eff_2) \wedge maxskip\ ;\ \Diamond g_1$ | |
|---|---|
| Step | $P'$ |
| $ex(a_2)$ | $\Diamond g_1$ |

$P$

| $\Diamond g_0\ ;\ \Diamond g_1$ | |
|---|---|
| Step | $P'$ |
| $\circ false$ | $\Diamond g_0\ ;\ \Diamond g_1$ |

The first list explains the simplest task of plan interpretation: if the plan begins with a concrete action then the robot's next step is to perform this action. The refinement process also checks the applicability of the action in the current state of the world, i.e. it proves that $S \rightarrow pre_{a_1}$ is satisfied. $P'$ will be scheduled to be the input for the next plan interpretation cycle.

The second list describes the case where a concrete action must be chosen to reach an immediate subgoal, we assume that $eff_1$ doesn't hold in the current state $S$. The action chosen to refine the specified problem has not to be necessarily the same as proposed during plan construction time. Possibly, that action is not applicable in the current

state $S$. The rest of the plan, $P'$, is an abstract goal which is scheduled to be a new input for the deliberative planner after the action $a_1$ has been really executed.

In the third list we see how a benevolent event is handled. Assuming that the subgoal $\mathit{eff}_1$ already holds in the current state the robot doesn't have to execute an action to reach it explicitly (the temporal logic formula *maxskip* is refined to describe an interval of length zero). Further assuming that $\mathit{eff}_2$ doesn't hold in state $S$ we act as described w.r.t. the second list. The refinement strategy used to solve the plan interpretation problem prefers the "null-plan" refinement against the "explicit-action" refinement.

The fourth list shows the null-plan refinement used when only abstract goals remain to be interpreted. They directly initiate a new deliberation step by the planner.

The planner is also activated when the action proposed to schedule for execution is not applicable in the current state $S$. There are at least two options what the new goal for the planner should be about: it can specify that a repair plan should be found in order to be able to execute the original action or to reach the immediate subgoal, e.g. w.r.t. the second list the goal for the planner is abstracted as $\Diamond \mathit{eff}_1 \; ; \Diamond g_1$. The new goal can also specify that deliberation should begin (partially) from scratch, e.g. the original goal is used, but now starting from other prerequisites; in the example above it is the goal $\Diamond g_1$. Comparable problems arise when preservation constraints contained in a plan are violated. Their satisfaction is checked in the current state just as the applicability of the proposed action.

- A feasibility test
  Before an action proposed to be scheduled for execution by the plan interpreter can really be executed it must be checked whether its execution is conformable to the behavior constraints $SC$ of the robot. Since the world state could have been changed by events in various ways it is not guaranteed that the actions, which were conformable to the constraints during plan construction time, also satisfy these constraints w.r.t. the current state. The problem to be solved is e.g. specified as

  $$S \wedge ex(a_1) \wedge SC \rightarrow \circ SC$$

  when $a_1$ is the proposed action; the invariance of the constraints $SC$ w.r.t. the proposed action and the current state $S$ must be proved. A negative result of this test activates a replanning as described by the plan interpretation process above, but with some further option: there is the possibility of backtracking in the case that an action refinement is performed during plan interpretation, all alternatives of possible actions are tested before replanning starts.

- Symbolic plan execution
  If the feasibility test has been successfully completed the proposed action $a$ can be executed in the current state $S$. This is logically described by solving the problem

  $$S \wedge ex(a) \rightarrow \circ \mathsf{New\_S}, \quad \mathsf{New\_S} \text{ a placeholder}$$

  using an appropriate progressive instantiation of a's action axiom. The instantiation of New_S is a description of the new world state.
  Events described in the set E (see above) are also executed this way in the simulation environment.

## Conclusion

We describe how a modal temporal logic framework can be used to define a very flexible plan execution environment which is able to handle a lot of phenomena induced by dynamism in a simulated world. The individual processes of the execution system are all described by appropriate logical inference procedures. Requirements on incrementality and flexible refinement are satisfied in a natural way. The framework clarifies and simplifies the relations of arbitrary complex plan (problem) specifications and flexible plan structures to the underlying semantic model of planning and plan execution.

## References


Bauer, M.; Biundo, S.; Dengler, D.; Feibel, H.; Koehler, J.; and Paul, G. 1996. RAP–Reasoning about Plans - A Progress Report. In Sauer, J.; Günter, A.; and Hertzberg, J., eds., *Proceedings des 10. Workshop "Planen und Konfigurieren" (PuK-96)*, 193–204. infix. Also appearing in Workshop Notes of the ECAI'96 Workshop on "Cross-Fertilization in Planning".

Biundo, S.; Dengler, D.; and Koehler, J. 1992. Deductive planning and plan reuse in a command language environment. In Neumann, B., ed., *Proceedings of the 10th European Conference on Artificial Intelligence*, 628–632. Vienna, Austria: John Wiley & Sons.

Dengler, D. 1994. An adaptive deductive planning system. In Cohn, T., ed., *Proceedings of the*

*11th European Conference on Artificial Intelligence*, 610–614. Amsterdam, The Netherlands: John Wiley & Sons.

Dengler, D. 1996. Customized plans transmitted by flexible refinement. In Wahlster, W., ed., *Proceedings of the 12th European Conference on Artificial Intelligence*, 609–613. Budapest: John Wiley & Sons.

Manna, Z., and Pnueli, A. 1981. Temporal verification of concurrent programs: the temporal framework for concurrent programs. In *The Correctness Problem in Computer Science*. Academic Press.

Manna, Z., and Pnueli, A. 1992. *The Temporal Logic of Reactive and Concurrent Systems. Specification*. New York - Berlin - Heidelberg: Springer - Verlag.

Rosner, R., and Pnueli, A. 1986. A choppy logic. In *IEEE Symposium on Logic in Computer Science*, 306–314.