A Conceptual Framework for Post-Design Analysis in AI Planning Applications

Tiago Stegun Vaquero[1] and **José Reinaldo Silva**[1] and **J. Christopher Beck**[2]
[1]Department of Mechatronics Engineering, University of São Paulo, Brazil
[2]Department of Mechanical & Industrial Engineering, University of Toronto, Canada
tiago.vaquero@usp.br, reinaldo@usp.br, jcb@mie.utoronto.ca

Abstract

A disciplined design process does not guarantee a complete and flawless model of a planning problem, even with the existing methods and tools available in the AI community. A post-design project phase can identify essential hidden knowledge and directly impact the final planning system. The central theme of this paper is the design of a domain-independent framework to structure the post-design process of planning applications for a better understanding and capture of missing requirements. The framework feeds the re-modeling process, closing the loop of the design cycle. We describe the conceptual model of such post-design framework, named *Post-Design Application Manager* (postDAM).

Introduction

Since the end of the 1990s there has been an increasing interest in the application of AI planning techniques to solve real-life problems. In addition to characteristics of academic problems, such as the need to reason about actions, real-life problems require detailed knowledge elicitation, engineering, and management. It is necessary a systematic design process in which Knowledge and Requirements Engineering techniques and tools play a fundamental role. A well-structured life cycle to guide design increases the chances of building an appropriate planning application while reducing possible costs of fixing errors in the future. However, given the natural incompleteness of the knowledge, practical experience in real applications such as space exploration (Jónsson 2009) has shown that, even with a disciplined process of design, requirements from different viewpoints (e.g. stakeholders, experts, users) still emerge after plan generation, analysis and execution.

Conceptually, a design process does not guarantee a complete and flawless model of a planning problem, even with the existing methods and tools available in the AI community (McCluskey 2002; Vaquero et al. 2009). The identification of unsatisfactory solutions and unbalanced trade-offs among different quality metrics and criteria (Jónsson 2009; Rabideau, Engelhardt, and Chien 2000; Cesta et al. 2008) indicates a lack of understanding of requirements and preferences in the model. These hidden requirements and rationales raise the need for iterative model analysis, re-modeling and tuning process. However, the gathering and interpretation of the hidden requirements must be made carefully. The lack of a structured process for acquiring emerging knowledge can lead re-modeling to wrong directions and prevent a proper judgment of resulting plans, as well as their respective quality and tradeoffs.

The design process of planning domain models has a saturation point in which further model adjustment make no significant impact on the final plan. Therefore, the analysis and evaluation of generated plans - with respect to the requirements and quality metrics - becomes a fundamental step in the modeling cycle. Such a post-design process naturally leads to feedback and the discovery of hidden requirements for refining the model. While some hidden requirements are detected in a single plan analysis cycle, others are only discovery in after several analysis cycles; this aspect indicates that plan analysis have different stages of hidden requirements identification. In fact, the literature on knowledge engineering for planning has shown interesting tools and techniques for plan analysis, including plan animation (McCluskey and Simpson 2006; Vaquero et al. 2007), visualization (e.g. Gantt charts), virtual prototyping (Vaquero, Silva, and Beck 2010), and plan querying and summarization (Myers 2006). However, all that work did not consider different identification stages of hidden requirements and did not explore the effects of the missing knowledge in the re-modeling loop.

In this paper, we present a conceptual model of a post-design framework for AI planning applications, called *Post-Design Application Manager* (postDAM), that addresses the different stages of the plan analysis process, including (1) a short-term analysis stage with simulation and visualization of plans, (2) a long-term analysis stage with acquisition and re-use of human-centered rationales for plan evaluations, and (3) a stage for cross-project analysis that would provide generalization of knowledge from different domains to be applied in other planning application designs. The framework aims at structuring the post-design process for the continuous discovery and identification of hidden requirements (e.g., constraints and preferences) that would potentially improve the model and, consequently, the quality and performance of planners. postDAM was designed as a complement of the KE tool itSIMPLE (Vaquero et al. 2009) in order to support designer while dealing with post-design challenges.

This paper is organized as follows. First, we briefly in-

troduce the concepts of the postDAM framework, including its layered structure for handling the different stages of the plan analysis process. Then, we describe the layers of the framework and their role in discovering missing and hidden requirements, as well in the re-use of domain knowledge. We provide a short discussion about the components of the framework that have been implemented in itSIMPLE and the ones that should be addressed next. Finally, we conclude and provide some final remarks.

The postDAM Framework

Due to the challenge of capturing requirements from different sources during design and the natural incompleteness of knowledge in complex planning applications, domain models might not reflect exactly the behavior expected by developers and stakeholders. Besides the initial expectation reflected in the requirements, design participants tend to learn and recognize their real needs during the development of the artifact. Incompleteness can mean that most of the recognition process must be part of the post design process, where a better and practical view of the system is available. The *Post-Design Application Manager* framework, postDAM, aims to support such an essential phase of plan development.

The primary goal of postDAM is to structure the post-design process in order to organize the plan analysis and support the capture of emerging requirements. The post-design process was designed as a domain-independent integrated framework with three layers of plan analysis, each requiring a different level of abstraction. As illustrated in Figure 1, the three layers are: *Short-term Plan Analysis*, *Long-term Plan Analysis*, the *Cross-project Re-use Analysis*. The abstraction level in these layers ranges from single plan evaluation (for acquiring metrics and rationales and to identify missing requirements) to the analysis of several cumulative plan evaluations (to discover hidden knowledge and to allow a cross-domain re-use of rationales). Figure 1 illustrates the layered analysis and how the layers interact and feed the re-modeling process in order to close the design loop.

We do not aim to replace the analysis done during the initial design process, but to complement the knowledge acquisition process, enhancing the feedback provided by post-design. This complementary process is as important as those performed during model design.

In what follows, we describe the main goal and role of each proposed layer of the framework.

First Layer: Short-term Plan Analysis

The first layer in the framework aims to analyze plans directly during the design and development phase. The goal is to detect malformed solutions and identify inconsistent behavior, missing requirements and misinterpreted features that can be clearly spotted by design actors. Each plan is analyzed individually in an attempt to verify the overall coherence of the model and of the resulting plan. Figure 2 illustrates the short-term analysis process.

As the first step in this layer, plans must be validated to guarantee that they are in fact solutions to the specified planning problem and that they do not violate any constraints or preferences defined in the domain model. Performing analysis without first checking plan validity is a waste of time and resources. If a plan is not a valid solution, it is necessary to revise the model of the domain, the plan itself, and the planner that generated it.

If a valid plan satisfies a problem specification, it does not mean that it can be directly applied and executed in real life. There are two main reasons why a plan cannot be directly executed. First, the plan might be composed of high level actions that can not be directly executed. This situation generally requires a post-processing to translate the high level plan into a low level sequence of actions or procedures that can be executed by controllers. Second, the specification of the planning problem and domain of application might be incomplete which would lead to unexpected results during the execution. For example, the lack of a safety constraint in the model to prevent robots from navigating too close to each other might result in crashes during execution. In this case, the execution of the plan must be investigated seeking for possible incoherence and unexpected characteristics. However, testing plan execution in real word can be very costly, risky, and sometimes impossible. Thus, a more elaborated plan analysis must take place. The postDAM framework uses a Virtual Prototyping (Cecil and Kanchanapiboon 2007) approach for such analysis.

A virtual prototyping stage is included in the first layer as a mechanism for plan simulation, which can reveal incompleteness and omissions in the problem specification or in the model of the domain of application. The virtual prototyping approach provides several advantages in system design (Cecil and Kanchanapiboon 2007). The approach has been widely used in industry for reasons such as: it provides cross-functional evaluation at a lower cost; it enables engineers to consider costly mistakes and downstream issues earlier in the product design cycle; and it facilitates better communication of product design issues among engineers of varying backgrounds (Cecil and Kanchanapiboon 2007). Connecting AI planning with the literature on virtual prototyping would benefit from these advantages.

The purpose of this visual technique is twofold. Firstly, it serves as a mimic of the real world in which the plan will be executed. The execution can consider physical properties that illustrate the applicability of plans in real scenarios. Properties such as collisions, gravity, mass, inertia and others can be verified (most of the available development environments for virtual prototyping have these properties already implemented in embedded engines - they can be included or not in the virtual model). Second, the visual and sound effects provide an excellent means of communication among design actors, such as domain experts, planning experts, stakeholders and users. These effects can give experts and non-experts a clear view of the domain model as well as an underlying planning strategy, as opposed to looking at plan traces. Flaws and missing constraints, not detected during design (or specification), can be spotted and discussed by visual inspection (Cecil and Kanchanapiboon 2007). Hypothetical examples of model inconsistency are: a robot trespassing solid bodies or sharp corner areas; a robot's battery reaching undesirable or unfeasible power levels; or a (pro-

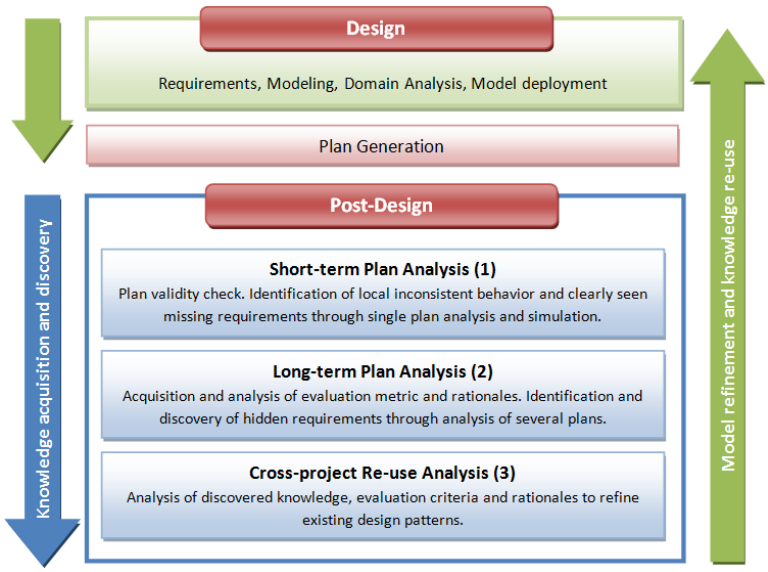

Figure 1: The conceptual model of the postDAM framework

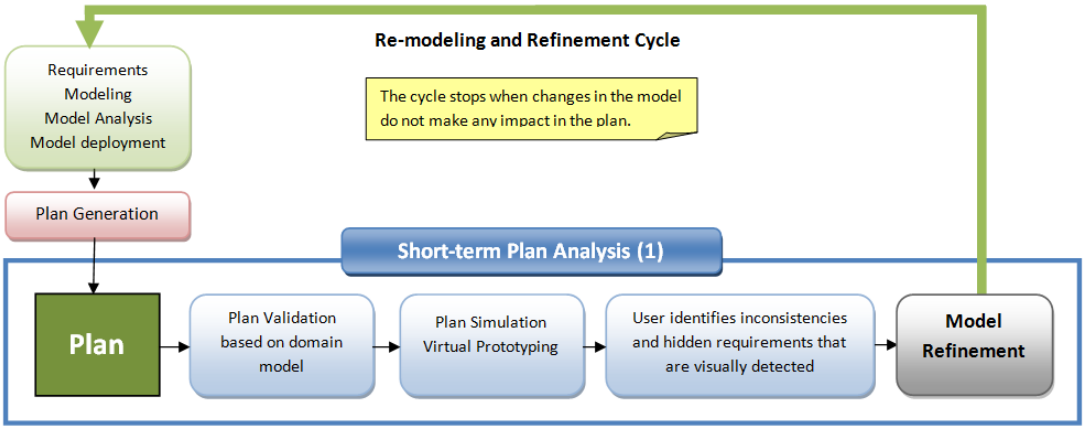

Figure 2: Short-term analysis in the postDAM framework

hibited) sequence of movements made by an autonomous air vehicle. Generally all these behaviors lead to a revision in the domain model. Besides the advantage in the model communication, a visual approach enriches the understating and familiarity with designs and resulting plans. Depending on the feedback provided during this analysis, early design phases must also be revisited.

Other plan visualization and simulation techniques would also fit on this layer of the framework (e.g., charts and diagrams). Similar to virtual prototyping, they serve as plan communication and plan analysis in the identification of inconsistencies in the model. However, we believe that virtual prototyping has a powerful communicative aspect via the visual interaction.

Creating virtual prototypes demands time and consumes resources, especially if applied to large and complex systems. However, its advantages in design support and decisions pay off, as seen in many real engineering problems (Cecil and Kanchanapiboon 2007). In addition, not all planning applications can be represented in visualization environments. For example, clinical decision support systems for oncology therapy planning (Fdez-Olivares, Cózar, and Castillo 2009) do not have a clear virtual representation to support analysis. For these cases, system or software prototyping becomes necessary to provide the same communication of the model for inconsistency detection.

The short-term layer and the re-modeling cycle are executed as many times as necessary. The cycle stops when there exists a minimal degree of acceptance (validation) from the design actors perspective and when changes in the model do not make any impact on the generated plans. After the short-term analysis, the generated plans are considered

valid, satisfying the goals and constraints specified in the model and having no inconsistencies or flaws.

In order to support the process described in Figure 2, the postDAM framework integrates a knowledge engineering tool (itSIMPLE (Vaquero et al. 2009)) for the modeling and re-modeling processes with a virtual prototyping environment. The former must provide a structured and disciplined design process while the latter must have a rich simulation environment that allows realistic representation of the real domain of application. In this work, we focus on domain-independent tools that are able to represent and model a variety of planning applications. It is worth to note that existing work and contributions in plan verification & validation, and plan execution & control (e.g. (Giannakopoulou et al. 2005; Fox, Howey, and Long 2005; Cesta et al. 2008)) could enhance, complement or fulfill the requirements of the short-term level, depending of the application.

Second Layer: Long-term Plan Analysis

In real planning applications, distinct valid plans can solve the same planning problem; however, they might have different qualities and strategies. In some applications (such as space exploration (Cesta et al. 2008)) there are a number of criteria to evaluate a plan and complex trade-offs to be achieved. The definition of a quality metric must be made wisely and carefully in AI planning problems. The use of the appropriate quality metrics and the consideration of evaluation rationales are essential to the selection of a subset of valid plans that can be reliably executed. In real applications, trade-offs must be known and acknowledged.

The second layer of the postDAM framework aims to support plan evaluation and model refinement based on a set of specified metrics. Plan evaluation involves the analysis of quality metric values, plan classification, and rationale acquisition. The successive plan evaluations gathered in the long-term can provide important information that can be pieced together, accessed and explored. This information is the base for discovering hidden requirements, constraints, preferences and the real intentions of design actors (knowledge that was not identified during the virtual prototyping phase). Such discoveries feed the model refinement cycle. Differently from the first layer, the model refinement envisaged in this long-term analysis focuses on enhancing the plan quality as opposed to plan validity. Figure 3 illustrates the long-term plan analysis process proposed in the framework.

As illustrated in Figure 3, the first step in the layer refers to an initial plan evaluation based on a specified set of quality metrics. In this initial evaluation, the goal is to analyze the values of each metric, their weights (representing the importance among other metrics) and to provide a classification for each one of them (e.g., a satisfiability level ranging from bad to good, 0 to 1).

Depending on the specification and experts' familiarity with the domain, it is possible that some of the metrics might have predefined classification functions that map metric values to classification ranges. The work of (Rabideau, Engelhardt, and Chien 2000) describes an interesting approach for representing predefined metric classification functions in which each metric is attached to a preference function that maps values to scores in the interval [0,1] (0 = unsatisfactory, 1 = satisfactory). This approach is used as a reference in the framework for representing predefined classification of quality metrics. If such predefined metric classification is available (for instance through a KE tool), the initial evaluation of the metrics can be made automatically. However, in most real scenarios predefined metrics functions are unknown. In fact, the complete set of necessary quality metrics might be unknown in advance. The lack of a complete set of metrics requires design participants to communicate and discuss the main aspects of their planning problem solutions during plan analysis.

As a second step, the framework introduces the process of acquiring plan metrics, plan classifications and plan evaluation rationales. Since the set of quality metrics might be incomplete, users can specify new ones during the initial plan evaluation. Based on individual metric classification and on the overall characteristics of the plan, design actors provide the final plan classification by using, for example, the interval [0,1]. Both metric and plan classification are done interactively with users.

Besides plan classification, the framework aims to acquire human-centric evaluation rationales. Different from existing work on rationales in planning (Polyank and Austin 1998; Wickler, Potter, and Tate 2006), we focus on acquiring human-centric rationales that emerge from user feedback, observations and justifications during plan evaluation. Based on general and individual criteria, interests, feelings and expectations, the rationales from plan evaluation generate explanations and justifications as to why a plan was classified into a specific quality level. Therefore, we extend here the concept of plan rationales described in planning literature with rationales that encompass "why a certain plan element does not or does satisfy a criterion" or "why a certain plan does not or does satisfy a preference". Moreover, these rationales could explain "why a certain metric does not or does satisfy a given criteria" and "what is the effect of a given plan characteristic or element in the plan quality" (e.g., it decreases or increases the quality). As an example of plan rationale, one might say that "the plan has a decreased quality because the robot left a block too close to the edge of the table" or "the plan has a high quality because the robot avoided repeatedly passing through the two most crowded areas of the building while cleaning it". We call these explanations *plan evaluation rationales*.

As illustrated in Figure 3, the acquired rationales for plan evaluations are checked to guarantee correctness and applicability to the plan being evaluated. Valid rationales are attached to the plan. Once analyzed and evaluated, the plan is then stored in a database of the postDAM framework, called *Plan Analysis Database*.

The *Plan Analysis Database* is an essential component of the postDAM framework and, in particular, of the long-term layer. The main role of the database is to support the reuse of rationales in each initial plan evaluation, i.e., the framework is responsible for accessing the database and looking up for rationales that can be applied. For example, if a previously analyzed plan in the database was annotated with a given

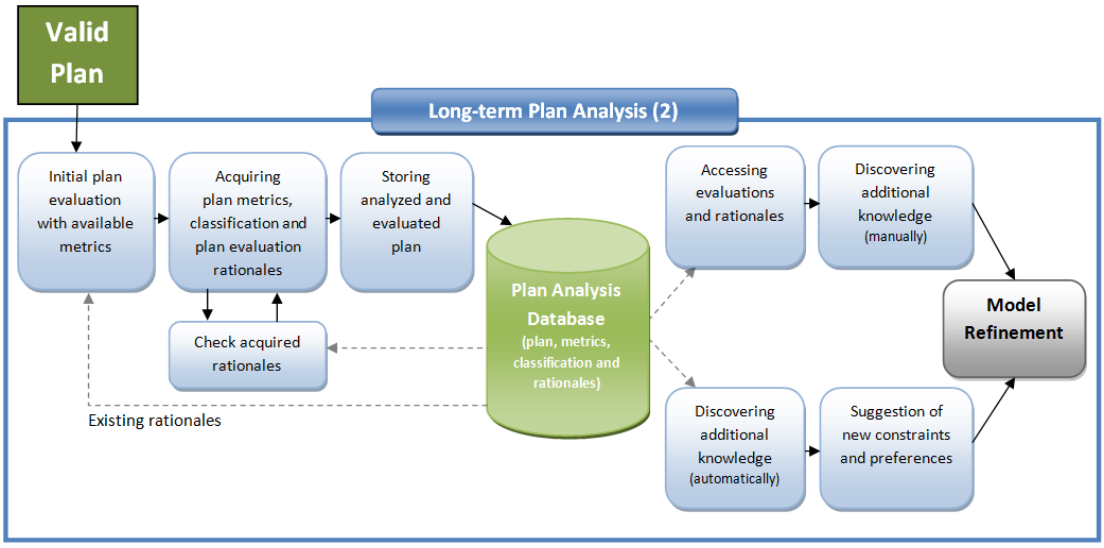

Figure 3: Long-term analysis in the postDAM framework

rationale that refers to a bad sequence of actions and if the plan currently being analyzed has the same bad sequence, the rationale can be reused to justify an initial classification during the evaluation.

Another role of the database is to support knowledge discovery in the long-term scenario. In the right side of Figure 3, we have split the exploration of the analysis data into two processes. The upper process refers to the manual access to the database and discovery of hidden requirements, preferences and constraints. The lower process refers to the automatic reasoning about plan evaluations and discovery (or extraction) of the hidden knowledge. Both processes address knowledge that directly impact in the quality of plans.

The manual process includes the access of the database to explore and study the information available from several plan analyses. A proper interface for accessing the data is required. By investigating and comparing different plans, classifications, and rationales, designers can perform modification to the model as they better understand actors' preferences, justifications and the correlations between a plan's properties and classifications.

Since manual identification of hidden requirements can be time consuming (but sometimes very efficient), an automatic process for knowledge extraction and suggested model adjustments becomes necessary. The postDAM includes such automatic process which can include specially designed algorithms and AI techniques. We do not get into the discussion of any algorithm or technique in this paper (an example is given in (Vaquero 2011)). Nevertheless, the input and output must be made clear. The input is a set of evaluated plans (along with their respective classification and rationales) and, if necessary, a domain and problem description. The output is a set of model modification suggestions to be analyzed by designers (e.g., a new precondition, postcondition, constraint, preference, metric, or action).

In order to support the process described in Figure 3, the postDAM framework integrates the following: (1) the knowledge engineering tool itSIMPLE for plan evaluation (including metric acquisition, specification and classification), rationale acquisition, and manual discovery of hidden requirements; (2) a database for storing analyzed plans; and (3) AI algorithms and techniques for reasoning about plan evaluations and discovering model refinements. Besides these elements, it is necessary to consider a good representation language to capture plan structures, metrics, classification and rationales. According to (Polyank and Austin 1998), the most reasonable approach would be to consider new and existing extensions of Plan Ontologies. In this work, we propose the use of an ontology-based representation of plans and rationales to allow reasoning about plan evaluations. Such an ontology is an extension of existing plan ontologies that is able to support post-design analysis. The proposed ontology-based representation of plans and rationales is discussed in the implementation section.

Third Layer: Cross-project Re-use Analysis

The two previous layers were designed to analyze a particular domain model. However, the experience gained in the design of one application can be applied to others with similar properties. The cross-project layer aims to abstract some of the experience gathered in the short and long-term analyses and make them available and reusable in the design of new planning applications. In this section, we focus on the reuse of quality metrics, rationales for plan evaluations and discovered knowledge.

Reusing past experience during design has long been used by other engineering areas, such as Software Engineering. The term *design pattern* is commonly used to refer to elements of reusable software. A design pattern is a general, reusable solution to a commonly occurring problem. Design patterns have been successfully used in software engineering to speed up design, improve quality, reduce cost, and

decrease problem fixing issues (Gamma et al. 1995).

Even though research on design patterns for AI Planning and Scheduling is scarce, studies like (Long and Fox 2000; Simpson et al. 2002) have shown that reusable elements can indeed be useful. Initial work on design patterns for Planning and Scheduling focuses on enhancing automated planners to produce plans faster through recognition of common patterns and the use of specialized heuristics (Long and Fox 2000; Clark 2001). The only work that focuses on the advantages of reusable knowledge during the design process is (Simpson et al. 2002). The most significant existing design patterns are: *transportation* encapsulating the behavior of mobile objects that traverse a network of locations (Long and Fox 2000); *construction* representing the behavior of objects that are built from other objects (e.g., assembly problems) (Clark 2001); and *bistate* referring to the behavior of on-off switches (Simpson et al. 2002). The existing design patterns are limited to classical problems, but this topic has great potential to evolve with richer domains.

Since design patterns already provide a structured foundation for reusing past experience, in the cross-project reuse analysis layer we focus on enhancing the patterns for planning with the knowledge gathered in previous plan evaluations performed in the short and long-term layers. Figure 4 illustrates the process proposed in the layer.

discussed

The main components of the layer described in Figure 4 are the *Plan Analysis Database* and the *Planning Design Patterns Database*. The former stores specific experiences acquired in plan evaluation cycles (in different domains), whereas the latter stores general encapsulated, reusable model elements for planning problems.

The first step represented in Figure 4 aims to generalize specific knowledge found in the plan analysis database and enrich the elements in the design patterns database. In this work we focus on making rationales and discovered knowledge available in design patterns. A matching process between design patterns and the existing knowledge from the plan analysis database is required. Conceptually, when a pattern is recognized in a particular domain model that has its plan analysis stored in the plan analysis database, the rationales, metrics and model modification referring to that particular pattern can be analyzed and transfered to the design patterns database. Since the plan analysis database can be highly dynamic (new data being stored in each analysis) an update cycle is necessary to filter new experiences and maintain the design patterns database. The techniques and methods for recognizing and updating design patterns are not in the scope of this paper. However, ontology matching techniques could be useful in such process (Euzenat and Shvaiko 2007). In fact, the use of an ontological representation of evaluated plans and rationales in the second layer provides a foundation for the application of such ontology matching techniques.

When a new design takes place for a planning application, existing patterns can potentially be used and explored. The framework is responsible for providing a design pattern catalog during the modeling phase. When designers import a selected pattern, all information can be re-used (e.g., quality metrics, preferences, additional constraints, and evaluation rationales). Design patterns might also be applied in existing models. In these cases, the framework is also responsible for identifying existing patterns in the model, matching the available knowledge and bringing the attached experience, if it is not yet explicitly specified in the model.

In order to support the process described in Figure 4, the postDAM framework conceptually integrates the following: (1) the knowledge engineering tool itSIMPLE for performing the update cycle on the planning design patterns database and for providing such patterns during application design; (2) a database of design patterns; and (3) mechanisms and techniques for recognizing and matching model components. Note that it is necessary to consider formalisms and languages for representing and reasoning about design models. The KE tool has an important role to provide the right reusable information at the correct phase of the design.

Implementation in itSIMPLE

In order to support the different layers of plan analysis, the postDAM framework has been implemented as a post-design tool for AI planning that integrates several tools, including an open-source KE tool, an open-source 3D content creation for virtual prototyping, an ontology-based reasoning system, and a database. The core of this integration is the KE tool, itSIMPLE (Vaquero et al. 2009). The KE tool has an important role in every layer of the framework; it supports human interactions, from metric and rationale acquisition to knowledge discovery and re-modeling. We have been designing a series of new extensions of itSIMPLE to fulfill such roles.

Currently we have completely implemented the first layer of the framework and partially implemented the second layer. Regarding the first layer, we have implemented an extension of itSIMPLE, integrating the KE tool with the plan validation VAL (Howey, Long, and Fox 2004) for plan verification and a virtual prototyping environment called Blender[1] for plan simulation. The work described in (Vaquero, Silva, and Beck 2010) provides a complete description of the short-term plan analysis process. The work introduces a re-modeling support tool that uses virtual prototyping techniques for plan analysis. While observing plan execution and evaluation in a 3D environment environment, users detect inconsistencies, unexpected behaviors and hidden requirements that guide a re-modeling process. In (Vaquero, Silva, and Beck 2010) we show, through experiments with benchmark domains, that the impact of a re-modeling tool on the planning process can be quite impressive, affecting not only the plan quality, but run-time and solvability. In these experiments, we have detected missing constraints on the definition of operators, as well as new predicates which were added to avoid specific scenarios. Such experiments open a large road for the investigation and understanding of knowledge captured on post-design.

In the long-term plan analysis, itSIMPLE (Vaquero et al. 2009) has also a central role; it supports human interactions and reasoning processes, from plan evaluation to ratio-

[1] Blender, available at www.blender.org.

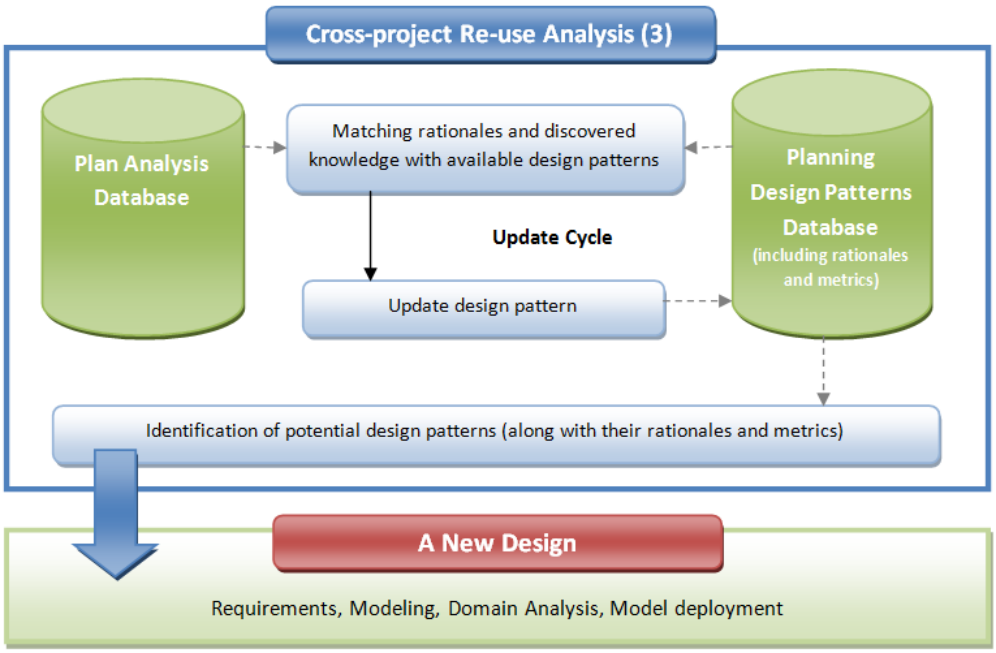

Figure 4: Cross-project re-use analysis in the postDAM framework

nales acquisition and knowledge extraction. We have implemented the following processes: the interactive plan evaluation in which users are able to modify and adjust plan classification based on their impressions; and the acquisition and re-use of plan evaluation rationales. This implementation is describe in the work (Vaquero, Silva, and Beck 2011).

In order to give here an overview of the implemented components of the second layer, we have integrated itSIMPLE with a database system, PostgreSQL,[2] and with a reasoning system, based on tuProlog.[3] The database is responsible for storing plan evaluations and their respective rationales, while the reasoning system is responsible for inferring rationale re-use in new plan evaluations. In order to capture, represent and re-use the rationales, we have used an extension of the Process Specification Language (PSL). PSL is an expressive ontological representation language of processes (plans), including activities and the constraints on their occurrences. PSL has been designed as a neutral interchange ontology to facilitate correct and complete exchange of process information among manufacturing systems such as scheduling, process modeling, process planning, production planning, simulation, project management, workflow, and business process applications (Grüninger and Kopena 2005).

We have not covered the development and investigation of all three layers of plan analysis and knowledge re-use. We have focused on the short and long-term analyses while the third layer is left for future work.

[2]PostgreSQL is available at http://www.postgresql.org/.

[3]tuProlog: see http://alice.unibo.it/xwiki/bin/view/Tuprolog/.

Conclusion

This paper presents the conceptual model of the postDAM framework. We have emphasized the structure of the proposed framework, i.e., its different layers of plan analysis that provide a basis for the model modification and refinement. Such refinement aims at the improvement of plan quality and, as a consequence, the planning performance. The layers of analysis include a short-term support for detection of missing requirements, a long-term support for identification of hidden knowledge, a cross-project analysis and software tools for knowledge re-use.

We have described the different tools that can be integrated in the framework to assist the discovery of missing requirements, to support evaluation rationale acquisition and re-use, and to guide the model refinement cycle. Not all processes have been implemented, but some of them have been developed. In previous work we implemented for example the first layer of postDAM. We demonstrated in (Vaquero, Silva, and Beck 2010) that following a careful post-design analysis, we can improve not only plan quality but also solvability and planner speed. However, in this paper we have shown the big picture of our research on post-design, aiming at supporting the different stages of plan analysis. In a real planning application, the analysis process that follows design becomes essential to have the necessary knowledge represented and adapted in the model.

References

Cecil, J., and Kanchanapiboon, A. 2007. Virtual engineering approaches in product and process design. *The International Journal of Advanced Manufacturing Technology* 31(9-10):846–856.

Cesta, A.; Finzi, A.; Fratini, S.; Orlandini, A.; and Tronci, E. 2008. Validation and verification issues in a timeline-based planning system. In *Proceedings of the 18th International Conference on Automated Planning and Scheduling (ICAPS 2008) Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Clark, M. 2001. Construction domains: A generic type solved. In *Proceedings of the 20th U.K. Planning and Scheduling Workshop*.

Euzenat, J., and Shvaiko, P. 2007. *Ontology matching*. Heidelberg (DE): Springer-Verlag.

Fdez-Olivares, J.; Cózar, J.; and Castillo, L. 2009. OncoTheraper: Clinical Decision Support for Oncology Therapy Planning Based on Temporal Hierarchical Tasks Networks. In Riaño, D., ed., *Knowledge Management for Health Care Procedures*, volume 5626 of *Lecture Notes in Computer Science*. Berlin, Heidelberg: Springer-Verlag. 25–41.

Fox, M.; Howey, R.; and Long, D. 2005. Exploration of the Robustness of Plans. In *Proceedings of ICAPS 2005 Verification and Validation of Model-Based Planning and Scheduling Systems workshop*.

Gamma, E.; Helm, R.; Johnson, R. E.; and Vlissides, J. 1995. *Design Patterns: Elements of Reusable Object-Oriented Software*. Reading, MA: Addison-Wesley.

Giannakopoulou, D.; Pasareanu, C. S.; Lowry, M.; and Washington, R. 2005. Lifecycle Verification of the NASA Ames K9 Rover Executive. In *Proceedings of ICAPS 2005 Verification and Validation of Model-Based Planning and Scheduling Systems workshop*.

Grüninger, M., and Kopena, J. B. 2005. Planning and the Process Specification Language. In *Proceedings of ICAPS 2005 workshop on the Role of Ontologies in Planning and Scheduling*, 22–29.

Howey, R.; Long, D.; and Fox, M. 2004. VAL: Automatic Plan Validation, Continuous Effects and Mixed Initiative Planning Using PDDL. In *ICTAI'04: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence*, 294–301. Washington, DC, USA: IEEE Computer Society.

Jónsson, A. K. 2009. Practical Planning. In *ICAPS 2009 Practical Planning & Scheduling Tutorial*.

Long, D., and Fox, M. 2000. Automatic Synthesis and use of Generic Types in Planning. In *In Artificial Intelligence Planning and Scheduling AIPS-00*, 196–205. Breckenridge, CO: AAAI Press.

McCluskey, T. L., and Simpson, R. M. 2006. Tool support for planning and plan analysis within domains embodying continuous change. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling (ICAPS 2006) Workshop on Plan Analysis and Management*.

McCluskey, T. L. 2002. Knowledge Engineering: Issues for the AI Planning Community. *Proceedings of the AIPS-2002 Workshop on Knowledge Engineering Tools and Techniques for AI Planning. Toulouse, France* 1–4.

Myers, K. L. 2006. Metatheoretic Plan Summarization and Comparison. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling (ICAPS-06)*. Cumbria, UK: AAAI Press.

Polyank, S., and Austin, T. 1998. Rationale in Planning: Causality, Dependencies and Decisions. *Knowledge Engineering Review* 13(3):247–262.

Rabideau, G.; Engelhardt, B.; and Chien, S. 2000. Using generic preferences to incrementally improve plan quality. In *Proceedings of the Fifth International Conference on Artificial Intelligence Planning and Scheduling*. Breckenridge, CO.: AAAI Press.

Simpson, R. M.; Mccluskey, T. L.; Long, D.; and Fox, M. 2002. Generic Types as Design Patterns for Planning Domain Specification. In *Knowledge Engineering Tools and Techniques for AI Planning: AIPS'02 Workshop*.

Vaquero, T. S.; Romero, V.; Tonidandel, F.; and Silva, J. R. 2007. itSIMPLE2.0: An integrated Tool for Designing Planning Environments. In *Proceedings of the 17th International Conference on Automated Planning and Scheduling (ICAPS 2007)*. Providence, Rhode Island, USA: AAAI Press.

Vaquero, T. S.; Silva, J. R.; Ferreira, M.; Tonidandel, F.; and Beck, J. C. 2009. From Requirements and Analysis to PDDL in itSIMPLE3.0. In *Proceedings of the Third International Competition on Knowledge Engineering for Planning and Scheduling, ICAPS 2009*, 54–61.

Vaquero, T. S.; Silva, J. R.; and Beck, J. C. 2010. Improving Planning Performance Through Post-Design Analysis. In *Proceedings of ICAPS 2010 workshop on Scheduling and Knowledge Engineering for Planning and Scheduling (KEPS)*, 45–52.

Vaquero, T. S.; Silva, J. R.; and Beck, J. C. 2011. Acquisition and Re-use of Plan Evaluation Rationales on Post-Design. In *Proceedings of ICAPS 2011 workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Vaquero, T. S. 2011. *Post-Design Analysis for AI Planning Applications*. Ph.D. Dissertation, Polytechnic School of the University of São Paulo, Brazil.

Wickler, G.; Potter, S.; and Tate, A. 2006. Recording Rationale in <I-N-C-A> for Plan Analysis. In *Proceedings of the 16th International Conference on Automated Planning and Scheduling (ICAPS 2006) Workshop on Plan Analysis and Management*.