Using the Task Formalism Method to Guide the Development of a Principled HTN Planning Solution for the Construction Industry

Peter Jarvis

Artificial Intelligence Applications Institute
The University of Edinburgh
80 South Bridge, Edinburgh EH1 1HN
United Kingdom
Peter.Jarvis@ed.ac.uk

Graham Winstanley

Division of Computing
The University of Brighton
Lewes Road, Brighton BN2 4GJ
United Kingdom
G.Winstanley@brighton.ac.uk

Abstract

To develop a quality Hierarchical Task Network (HTN) planning application, one must understand how to use the representational devices provided by these systems to construct a principled model of an application domain. Support for this objective is currently limited to evolving guidance frameworks and repositories of domain descriptions. To date, there has been no description of the application of these guidance frameworks to form concrete domain descriptions. The developer is instead left to discover for themself the mapping between the steps of the frameworks and entries in the repositories. In this paper, we address this issue by describing the development of a HTN solution for the construction industry with the guidance of the TF Method. From this experience, we conclude that the TF Method offers significant assistance to the knowledge engineer. Specifically, the method highlights the importance of a planned approach to the development of an application, a conscious commitment to a primary modelling method, and the gradual and considered development of HTN descriptions. The method, however, would benefit from pointers to complementary techniques developed in other areas (e.g. the KADS methodology), and from tool-support.

Keywords: AI Planning, HTN Planning, Knowledge Acquisition, Planning Domain Modelling.

1 Introduction

To develop a quality Hierarchical Task Network (HTN) [Tate 1977] planning application, one must understand (i) the representational devices provided by these systems and (ii) how to use those devices to construct a principled model of an application domain. The difference between points (i) and (ii) is similar to that between knowing how to write computer software (programming) and knowing how to engineer computer software (software engineering). Point (i) is addressed well in the planning literature by both the manuals provided by the developers of HTN systems (e.g. [Tate et al. 1994a, Wilkins 1997]) and the papers that discus the various aspects of these systems (e.g. [Tate 1996, Yang 1990, Erol et al. 1993; 1994, Erol 1995, Kambhampati 1994]). Support for point (ii) is, in contrast, still in its infancy [Tate et al. 1998, Erol 1995].

There are currently two sources of support for aiding the development of principled planning-domain models: *guidance frameworks* and *repositories of domain descriptions*. Guidance frameworks, such as the Task Formalism (TF) Method [Tate et. al. 1998, Tate et. al. 1995] developed to support the O-Plan system [Currie & Tate 1991], pull together the experiences gained by AI Planning experts in coding specific domains. These experiences are presented as a process defining the stages that a development should pass through and the issues that should be considered at each stage. The repositories of domain descriptions allow a system developer to learn from encoding techniques applied to previous domains.

To date, there has been no description of the application of these guidance frameworks to form concrete domain descriptions. The developer is instead left to discover for themself the mapping between the steps of the frameworks and entries in the repositories. In this paper, we address this issue by describing the development of a HTN planning solution for the construction industry with the guidance of the TF Method. This description provides both a demonstration and an evaluation of the TF Method in an industrial context by people who were at the time of encoding independent from the O-Plan design team.

This paper is organised as follows. Section 2 briefly outlines the purpose of the encoding described in this paper and the context of its development. Section 3 describes how the TF Method's recommendation that a planned approach to the development of a domain description was realised with the assistance of the KADS methodology. Section 4 details how the TF Method's guidance on the writing of the description that will be input to a planner was followed. Section 5 draws conclusions about the utility of the TF Method in the light of the experiences described in this paper and identifies issues that future research should address.

2 Context of the Encoding

The encoding presented in this paper was produced to underpin a research project at The University of Brighton, UK. It was used to motivate and test a new planning architecture designed to combine the relative strengths of HTN and Model-Based Planning [Jarvis & Winstanley 1998; 1996a; 1996b, Jarvis 1997]. The elicitation, encoding, and evaluation process carried out at Brighton was performed in the construction industry over twelve months. The process included meetings with experts and observations at construction sites. The work built upon previous case studies developed during a collaboration between The University of Brighton and The Center for Integrated Facility Engineering (CIFE) at Stanford University [Winstanley & Hoshi 1993].

To date, the publications produced by the project have not discussed the role of the TF Method in this development.

3 Realising a Planned Development of the Domain Description

The TF Method recommends that the development of a domain description be planned to ensure that it does not grow haphazardly or inconsistently. The method makes two specific recommendations for achieving this goal. First, that those domain experts used in the development are grouped into one of two roles. Second, that an early and conscious choice is made between the two primary modelling approaches of action expansion and goal achievement. Each of these issues is considered in turn in the following subsections.

3.1 Stage 1: Understand and Scope the Problem

Drawing from the perspective of project management, the TF Method uses role identification to define the activities different domain experts should play in the development process. The central controlling role is identified as the **Domain Expert** who is in charge of managing the scope of the application and is responsible for providing the overall structure of the domain description. Individuals in the role of **Domain Specialist** are then assigned to particular parts of that structure which they then "fill-in" with detailed knowledge.

Whilst, through the domain expert role, the TF Method (importantly) identifies the need to both scope and structure in a domain encoding, it does not guide the achievement of these objectives. For further assistance, we turned to the KADS Methodology [Schreiber et al. 1993a] and the Organisation and Application models [Schreiber et al. 1993b] within its model set.

The KADS Organisational Model supports the capture of the socio-economic environment within which a Knowledge Based System (KBS) is to be fielded. The model contains constructs for developing a graphical description of the tasks and functions within an organisation together with information that can help predict the effect of a KBS system on that organisation. A simplified version of the organisational model produced for the construction industry is shown in figure 1. This model was developed in two stages. First, the overall structure was developed with a senior director in the role of domain expert. The domain expert was asked questions of the type: *what are the main activities within your organisation?*, *how do those activities interact?*, and *what external organisations do you interface with and why?* Second, individuals working within the units identified by the domain expert were used in the role of domain specialist to develop an understanding of the detailed operation of those units. Domain specialists were asked questions of the type: *what are the activities you perform?* and *what other specialists do you interact with and why?*

Constructing the model provided an understanding of three important facets of the organisation: (i) its overall structure and function, (ii) the areas either performing the planning function or interfacing to it, and (iii) the commercial motivation for intelligent planning support. Whilst developing this understanding, the different views of domain expert and domain specialists complemented one another. The expert understood the overall process and the relationships between each stage but not the detail of how each was performed. Each specialist understood the detail of his or her area but not the compete context in which he or she worked. Reconciling these two views added a beneficial cross-check to the modelling process. Mismatches were traced to one of two causes. Either the knowledge engineer had misunderstood a specialist's or expert's comments or an organisational problem had been encountered. In the former case, the mismatch provided a useful tool for prompting both specialist and expert to clarify their comments. In the latter case, the mismatch motivated the specialist and expert to meet and clarify their perceptions of the process they engaged in. This cross-checking benefit was obtained at all stages in the development process through these roles.

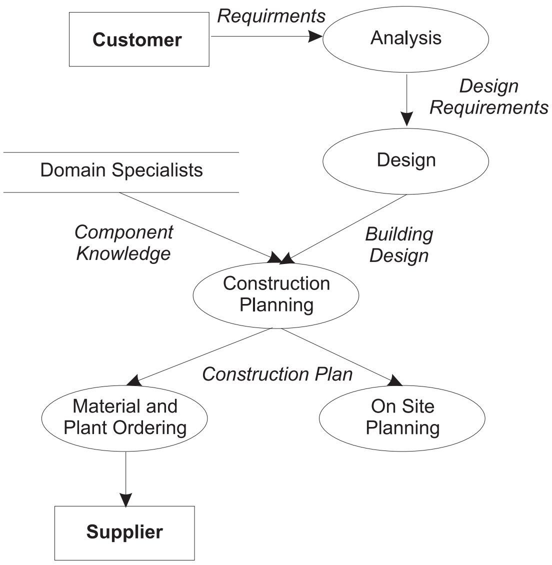

Figure 1: Organisation Model

The KADS Application Model supports the determination and definition of the scope of a KBS application. To develop the application model shown in figure 2, the same senior director was again used in the role of domain expert. The important decision made during the model's development was the selection of the stages from the planning process identified in the organisational model that would be supported by the AI Planning system. This decision was guided by two factors. First, developing the organisational model had identified project bidding as the area that would benefit the most commercially from intelligent tool support. When managing a bid, an organisation must consider the risk of spending resources on producing project plans for a project that may not be awarded. This risk must be offset against the chance of "successfully" winning a project by under estimating the cost of realising it through insufficient project planning. Considering figure 1, the bidding process activates the *analysis*, *design*, and *construction planning* processes to provide a skeletal outline of cost and timescale needed to construct proposed project. Second, the information technology infrastructure of the organisation was limited to its head office. Developing an application that included the on-site phases of the planning process would have required the introduction of computer equipment to the construction sites. The cost of this equipment and the associated training were considered prohibitive when compared with the potential cost saving from intelligent planning tools. These two factors lead to the decision to scope the application to supporting the project bidding aspect of the organisation's planning process. The resultant application model is shown in figure 2.

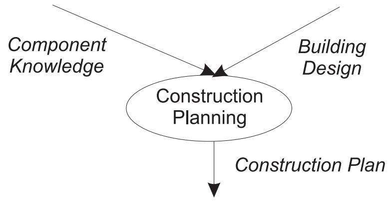

Figure 2: Application Model

In conclusion, the TF Method's emphasis on the need to scope and structure an application motivated the development of the KADS organisation and application models. The resultant models provided a clear understanding of the organisation and lead to an informed decision as to which aspects of it were to be addressed by the AI planning system. The separation of the user roles of expert and specialist mapped directly to the different views held by individuals in the domain. Reconciling the two views added a beneficial cross-check to the process.

3.2 Stage 2: Select either Action Expansion or Goal Achievement as the Primary Modelling Approach

The second stage of the TF Method recommends a conscious commitment to either action expansion or goal achievement as the primary (but not sole) modelling approach to a domain. This stage of the TF Method makes explicit that there are two modelling approaches and that if one mixes the approaches, one should do so consciously.

Experimental modelling with each approach was used to inform this decision. This experimentation categorised planning knowledge in the construction industry as being structured around the components of a building and the trades or specialists used to construct related groups of these components. A plumber, for example, is responsible for the installation of a building's bathroom fittings and a scaffolder is responsible for erecting the scaffolding that supports bricklayers in the task of constructing walls. This structure was readily mapped, as shown in figure 3, to the hierarchy of schemata inherent in the action expansion approach. Considering the earlier example, the overall task of installing a building's services was encapsulated within a single schema. This schema then refined to two schemata at a lower modelling level with the first describing the work of the plumbing specialist and the second the electrician specialist.

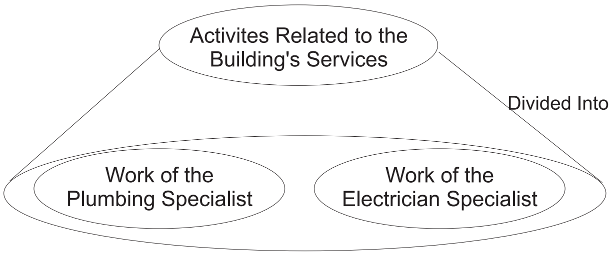

Figure 3: Decomposition of the Installation of a Building's Services by Trade

The goal achievement approach casts the construction problem in terms of a number of goals and a number of primitive actions that can be assembled to achieve those goals. The approach does not offer facilities for organising a domain description around into the "groupings" assigned to subcontractors nor the "levelling" of actions to allow a hierarchy of subcontractors. From this observation, action expansion was selected as the primary modelling approach with goal achievement as the secondary modelling approach. This "mixing" of the approaches is shown in *section 4.5*.

This experience with the Task Formalism provides evidence to support Drummond's thesis [Drummond 1994] that industrial planning problems are more readily addressed by action expansion than by goal achievement techniques.

4 Writing the Task Formalism Domain Definition

The third stage of the TF Method defines the process of writing the actual Task Formalism domain definition that will be input to the planning system. The stage is divided in the phases: (a) define the task definition, (b) define the schema hierarchy, (c) define the effects that will be produced by actions in the schema hierarchy, and (d) define the conditions that are required by actions in the schema hierarchy. Before performing each of these phases, it was considered beneficial to construct a model based upon the domain knowledge layer from the KADS methodology's expertise model. This model layer facilitates the identification and visualisation of the concepts, concept properties, and relationships between concepts in a domain. It was envisaged that constructing this model would give the knowledge engineer an understanding of the fundamental elements of the domain. This understanding would then be used to underpin the writing of the Task Formalism definition. It is this reasoning that lead us to add the stage "(3α) build a model of expertise" to the TF Method.

4.1 Stage 3α: Build a Model of Expertise

The application model shown in figure 2 identifies the inputs to the planning process as *component knowledge* and a *building design*. The expertise model developed in this section aims to identify expertise within these inputs in terms of concepts, concept properties, and relationships between concepts. To achieve this aim, we worked first with a senior director (in the role of domain expert) and a set of design drawings that described the structure of a building and the plan followed to construct that design. The design drawing corresponded to the building design input and the plan to the construction plan output. These artefacts were selected to help identify and describe the *component knowledge* input and understand how the construction process applies that knowledge to the *building design* input to produce the *construction plan* output. Once the overall structure of the expertise had been defined, other experts were used in the role of domain specialist to fill in the detailed knowledge. The benefits of using these two roles were similar to those described in section 3.1.

The primary domain concepts map directly to the components within a building and the concept properties to the properties of those components (e.g. physical dimensions, construction material, etc). Figure 4 shows an example of the definition that was produced for each concept.

concept *PILE*
properties:
length: mm
width: mm
height: mm
description:
Piles are used in areas where the ground is too soft
to support a building's weight directly. By driving a pile
into the ground, the weight of the building is supported
by the pile.
planning factors:
A pile must have its position set out accurately and a
pile mat must be laid to support the driving of a pile.

Figure 4: Example Concept Definition

Two distinct types of relationship between the concepts were identified:

- The *organisational relationships* are conceptual and allow experts to conveniently group components and therefore to reason at different levels of component aggregation. The organisational relationship, referred to as the *subcomponent* or *part-of* relationship, is represented by the diamond notation in figure 5. In the figure, the *building* component is shown as an aggregation of the components *plant equipment* through to a *roof.* These subcomponents of the building may be decomposed further. The *roof* component, for example, is an aggregation of the components *roof steel work* through to *roof covering.* Experts use these grouping to reason at different levels of abstraction. An expert assigning a contractor to the *roof* component, for example, would assume that this contractor would then be responsible for constructing all the subcomponents of the roof. The contractor may subcontract again elements of the roof's construction; however, from the perspective of the expert, the original contractor is responsible for the overall construction.

- The *dependency relationships* reflect physical constraints within a building's design that cause temporal ordering constraints between construction actions. Consider the *support* relationship between the *roof steelwork* and the *roof decking* shown in figure 5. The relationship represents the fact that the weight of the roof decking is physically supported by the steelwork. The decking must therefore be constructed after the steelwork.

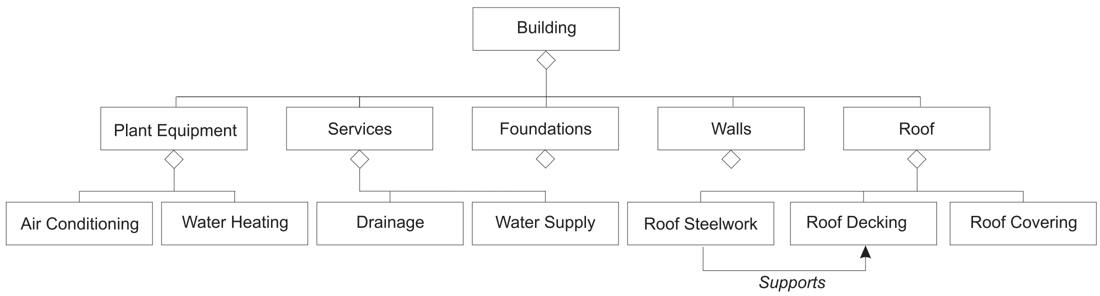

Figure 5: Concepts Augmented with their Structure and Dependency Relationships

The expertise model furnished the development process with a clear understanding of the concepts, concept properties, and relationships with the domain. This understanding permitted the application to be described as follows. The *component knowledge* input consists of knowledge about the components that may be included in a building, the properties components may hold, and the relationships that may exist between components. It includes knowledge about the actions required to construct each component and the effects of the relationships between those components in terms of temporal ordering constraints that must be placed between construction actions. The *building design* input is a configuration of the components with definite properties and relationships between components. The *planning process* applies the *component knowledge* to a specific *building design* to produce a construction plan.

4.2 Stage 3a: Define the Task Definition

This stage of the TF Method advises that the main task that the AI planner is to support be defined first. From the expertise model defined in figure 5, this task can be characterised as achieving the construction of a specific building design. This task was encoded in two parts and resulted in the task definition shown in figure 6. The *type statements* and *initially facts* define the components and relationships between components in a design. The single *task node* instructs the AI planner to develop a plan to construct the *building* component within the design. This encoding assumes that the remainder of the TF encoding will allow the planner to generate appropriate actions and ordering constraints for the subcomponents of the building.

```
types   plant_equipment = {plant_1},
        water_heating     = {water_heater_1},
        air_conditioning  =  {air_condition_unit_1},
        ...
initially {subcomponent  plant_1} =
          {water_heater_1, air_condition_unit_1},
        ...
task build_supermarket_extension;
   nodes 1 action {build supermarket};
end_task;
```

Figure 6: Initial Task Definition

It is important to note that the *type definitions* and *initially facts* correspond to the *building design* input to the planning process represented in the application model shown in figure 2. When changing the design the planner is to consider, the user should only have to modify these facts to express that new design. The remainder of the TF encoding should remain unchanged. In terms of the application model, the task definition corresponds to the *building design* input and the remainder of the TF encoding to the *component knowledge* input.

4.3 Stage 3b: Define the Schema Hierarchy

The role of the schema hierarchy is to define the actions that the HTN planner is to use to develop a plan to construct the building design defined in the task definition. The TF Method suggests that this hierarchy is developed by gradually working down from the high level actions to the more detailed actions. It is suggested that each level in the hierarchy be related to a modelling level in the application domain. The development of the expertise model described in section 4.1 discovered that planning knowledge in the construction industry is organised by domain experts and specialists into a component hierarchy. In keeping with the TF Method's advice, it was decided to use this component hierarchy to structure the schema hierarchy.

Figure 7 show a section of the component hierarchy and the modelling levels derived from it. The *task level* describes an overall building design and is mapped to the task definition encoded in section 4.2. The subcomponents of the building are assigned to the modelling level *aggregate components*. The term *aggregate* is used as each component is an aggregation of components defined at a lower modelling level. The lower modelling level is termed the *primitive level* as it contains components that are at the lowest level of abstraction to be considered by the planning process.

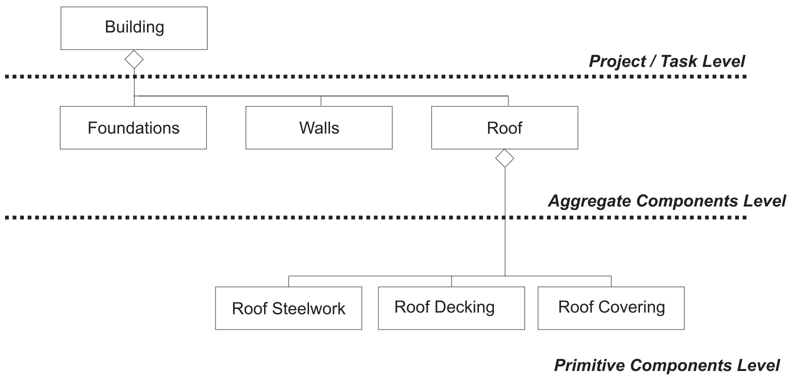

Figure 7: Modelling Levels Assigned to the Component Hierarchy

The first encoding challenge is to provide a schema that *expands* or *refines* the initial task definition to synthesise the actions that describe the components at the *aggregate components* modelling level. This challenge is meet by the schema *build_building* shown in figure 8. Consider the arrow that starts at the node statement in the task *build_supermarket_extension* and ends at the *expands* statement in the schema *build_building*. The arrow represents the schema expansion process performed by a HTN planner. It signifies that the *build_supermarket_extension* task will be refined to the actions included in the *build_building* schema. This encoding meets the requirements of the first challenge by ensuring that the components at the *aggregate components* modelling level will be considered by the planner.

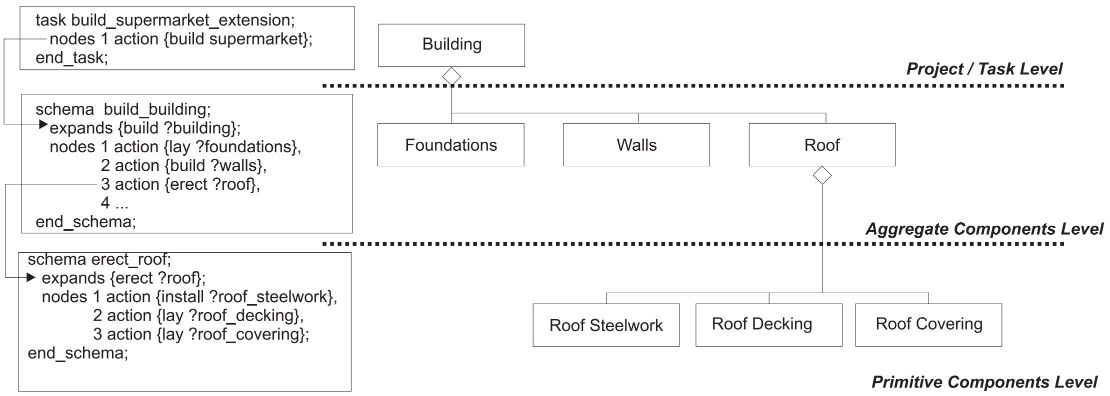

Figure 8: Schema Hierarchy in the Context of the Component Structure and its Modelling Levels

The second encoding challenge is to provide the schemata required to *expand* or *refine* the actions generated for the components defined at the *aggregate components* level to the include actions for the components defined at the *primitive components* modelling level. This challenge is met by providing a schema for each component at the *aggregate components* modelling level. Each schema includes the subcomponents of the aggregate component it represents. Figure 8 shows how this encoding scheme was applied to the *roof* component. Consider the arrow that starts at the *erect ?roof* node statement in the schema *build_building* and ends at the *expands* statement in the schema *erect_roof*. The arrow again represents the schema expansion process performed by a HTN planner. It signifies that the *erect ?roof* action will be refined to the actions included in the *erect_roof* schema. This encoding meets the requirements of the second challenge by ensuring that the components at the *primitive components* modelling level will be considered by the planner.

As shown in figure 8, this encoding scheme conforms to the advice in the TF Method by preserving the relationship between the modelling levels identified within the domain expertise and the HTN schema hierarchy.

4.4 Stage 3c: Define the Effects that will be Produced by Actions in the Schema Hierarchy

The TF Method suggests the effects produced by actions are assigned to modelling levels so to clearly define the statements about the world that will be manipulated at each level. Each component was considered in turn to determine the effect(s) that would result from its construction. The components at the higher modelling levels produce "aggregate" effects that describe the overall construction of their subcomponents. Constructing the *foundations* will, for example, add the effect *status_of_foundations = laid*. The components at the lower modelling levels produce effects that describe their own construction. Constructing the steelwork, for example, will add the effect *status_of_the_roof_steelwork = erected*.

Figure 9 shows the encoding of the action effects and their correlation to the modelling levels. The new TF elements are highlighted in bold font.

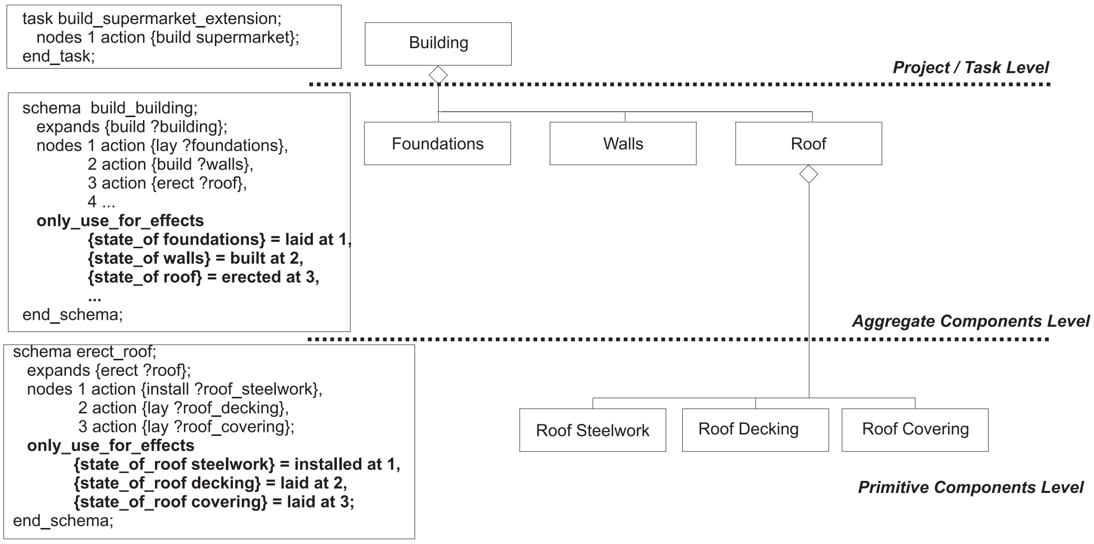

Figure 9: Encoding of the Effects Produced by Actions and their assignment to Modelling Levels

4.5 Stage 3d: Define the Conditions that are Required by the Actions in the Schema Hierarchy

The final stage of the TF Method offers guidance on the task on encoding the conditions required by actions. HTN planners offer a number of *condition types* that allow the domain writer to inform the system of the meaning of each condition in the application domain. The systems use this advice to select between the different mechanisms available for establishing and maintaining a condition. For condition typing to work effectively by avoiding the dangers of hierarchical promiscuity [Tate et al. 1994, Wilkins 1988], it is essential to carefully consider two factors. First, the relative positions in terms of modelling levels of a condition and the statements that can satisfy it. Second, the relationship between the encapsulation unit of the HTN schema and the schema or schemata within which a condition is placed and the statements about the world that can satisfy that condition. Tate et al. [1994] complement their TF Method by defining the constraints on the use of condition types in terms of these factors. Before considering action conditions, figure 10 was constructed to pull together the required information on effect levels and schema scope.

Figure 10 shows two dependency relationships as arrows which, as identified during the constriction of the expertise model in section 4.1, lead to temporal ordering constraints between actions.

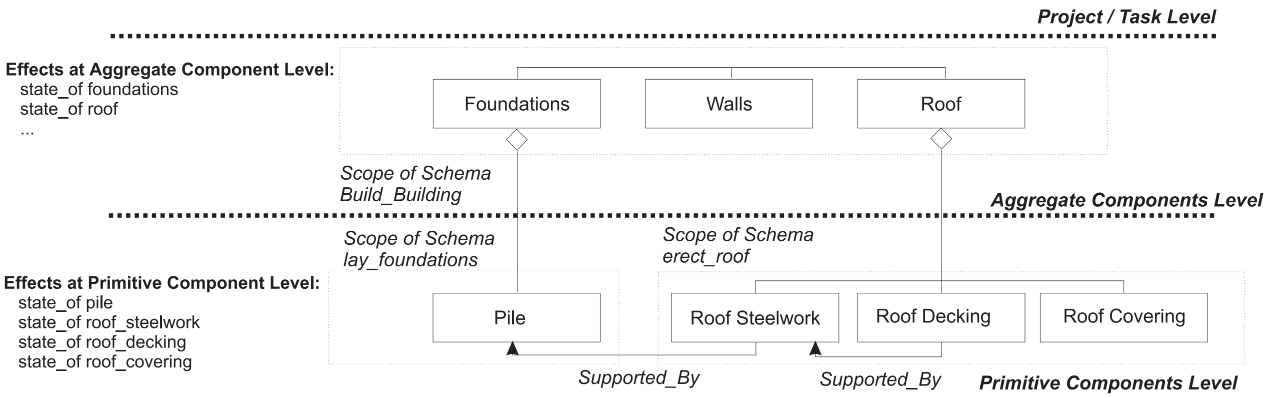

Figure 10: Scope of Schemata and the Modelling Levels at which Effects are Introduced

Consider the *support* relationship between the steelwork and decking components. In terms of the action effects written in section 4.4, this condition can be expressed as: *the status of the roof steelwork must be set to erected before the laying of the roof decking commences.* The scope information shown in figure 10 informs us of three factors relevant to the selection of an appropriate condition type. First, that this effect is produced at the *aggregate components* modelling level by the action *erect ?roof steelwork.* Second, that the condition is required by an action within the *aggregate components* modelling level. Third that the actions participating in this relationship are defined within the same schema, *schema erect_roof.* These factors meet the constraints on the use of the *supervised* condition type described in [Tate et al. 1994]. The condition was therefore encoded as shown in figure 11 within the schema *erect_roof* through the statement *supervised status_of_roof_steelwork = erected at 2 from [1].*

```
schema erect_roof;
  expands {erect ?roof};
  nodes 1 action {install ?roof_steelwork},
           2 action {lay ?roof_decking},
           3 action {lay ?roof_covering};
  only_use_for_effects
          {state_of_roof steelwork} = installed at 1,
          {state_of_roof decking} = laid at 2,
          {state_of_roof covering} = laid at 3;
  conditions
          supervised {state_of roof_steelwork} = installed at 2 from [1],
          unsupervised {state_of pile} = laid at [1];
end_schema;
```

Figure 11: Encoding of Action Conditions as their Types

Consider the *support* relationship between the pile and steelwork components. In terms of the action effects written in section 4.4, this condition can be expressed as: *the status of the pile must be set to laid before the erection of the roof steelwork commences.* The scope information shown in figure 10 informs us of three factors relevant to the selection of an appropriate condition type. First, that this effect is produced at the *aggregate components* modelling level by the action *drive ?pile.* Second, that the condition is required by an action within the components modelling level. Third, that the actions participating in this relationship are defined within different schemata. These factors do not meet the constraints on the use of the *supervised* condition type. Conditions of this type may only be place actions that occur within the same schema. The factors, however, do meet the constraints on the use of the *unsupervised* condition type. The condition was therefore encoded as shown in figure 11 in the schema *erect_roof* through the statement *unsupervised status_of_pile = laid at [1].*

5 Conclusion

To develop a quality HTN planning application, one must understand not only the operation of HTN representational devices but also how to combine them to write a principled model. By a quality application, we mean one that actually represents the application domain under consideration so to both produce quality plans and to allow the planning knowledge base to be maintained.

This aim is currently supported by evolving *guidance frameworks* such as the TF Method and *repositories of domain description.* To date, no published work has animated the use of the guidance frameworks to produce a domain description. In this paper, we have addressed this issue by describing to use of the TF Method to construct a principled model for an application in the construction industry. From this experience, we draw the following conclusions on the utility of the TF Method in an industrial context:

- The method highlights the importance of a planned and considered approach to the development of a domain description and the need to carefully scope the application under consideration. The benefits of these activities are well understood within the software-engineering field. Whilst the method does not offer any assistance in meeting these objectives, it did highlight there importance and prompted a search for suitable techniques. The KADS methodology's Organisation and Application models provided adequate support for achieving these objectives. The process of constructing them furnished the knowledge engineer with a good understanding of the domain and lead to an informed decision about the scope of the application. It would be useful if the TF Method itself pointed to these and other suitable techniques to save the knowledge engineer the effort of searching for them.

- The distinction between the roles of *domain expert* and *domain specialist* mapped well to the different views held by individuals in the construction industry. Reconciling the two views provided an additional cross-check to the modelling process.
- The recommendation of choosing between the primary modelling method to be used in a domain had two benefits. First, it made apparent that there are two distinct modelling methods available to the knowledge engineer. Making this distinction ensured that the knowledge engineer did not haphazardly combine the approaches. Second, it prompted the knowledge engineer to investigate the properties of each technique and then to make an informed decision as to which was most appropriate to the domain under consideration.
- The method moves to the development of the TF encoding too quickly. We found that introducing the additional step of constructing a KADS expertise model furnished the knowledge engineer with an understanding of the concepts in the domain and the relationships between those concepts. This understanding proved invaluable when writing the TF. It simplified the task of identifying modelling levels and provided an understanding of the factors that lead to temporal dependency constraints between actions.
- The decomposition of the encoding process(*task definition, schema hierarchy, action effects,* and *action conditions*) motivated the knowledge engineer to concentrate on each distinct facet of a TF representation. The constraints on the use of condition types provided a good understanding of their operation and allowed the engineer to use these constructs in a considered way, avoiding the pitfalls of hierarchical promiscuity. These guide-lines currently reside within a conference paper outside of the TF Method's description. It would again be useful to include these within the TF Method description to make their existence clear to the knowledge engineer. Pulling together and maintaining the information about an encoding that is necessary to conform to the constraints condition types was both laborious and time consuming. This activity would benefit from tool support that is capable of producing this information in a similar format to that shown in figure 10.

From these conclusions, we suggest two areas that should be considered by further research:

- The TF Method would benefit from the integration of development methods that have been designed in other fields. The activities of obtaining an overall understanding of a domain, scoping an application, and building a model of the domain's expertise, in particular, are addressed well elsewhere. This integration would benefit from viewing the TF Method as an AI planning specific phase that tailors the results of these methods to the needs of planning application development. Promising work emerging in this area is described in [Tate et al. 1998].
- Various stages of the TF Method require information about the modelling level at which TF elements are positioned and the scope of schemata. This information could be readily produced and maintained by computer-based tools. The provision of such tools would remove the current need to laboriously record and maintain this information with pencil and paper.

Acknowledgements

The authors would like to thank the employees of The Llewellyn Group of Companies, Sussex, UK for their assistance in the elicitation, modelling, and evaluation activities that underpined the research presented in the paper.

We would also thank John Levine and Austin Tate for their comments on this paper.

References

[Currie & Tate 1991] K. Currie, and A. Tate. O-Plan: the Open Planning Architecture. *Artificial Intelligence*, 51(1), North Holland. *available from: http://www.aiai.ed.ac.uk/~oplan/oplan/oplan-doc.html*

[Drummond 1994] M. Drummond. On Precondition Achievement and the Computational Economics of Automated Planning. *In:* C. Backstrom, E. Sandwall, eds. *Current Trends in AI Planning*, IOS Press, 6-13.

[Erol et al. 1993] K. Erol, J. Hendler, and D. Nau. Semantics for Hierarchical Task-Network Planning, *Technical Report CS-TR-3239, Computer Science Department, University of Maryland*, Maryland, USA.

[Erol et al. 1994] K. Erol, J. Hendler, and D. Nau. Task Refinement Planning: Complexity and Expressivity. *In: Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, Seattle, USA.

[Erol 1995] K. Erol. Hierarchical Task Network Planning: Formalisation, Analysis, and Implementation. *Phd Thesis*, Computer Science Department, University of Maryland, Maryland, USA.

[Jarvis & Winstanley 1998] P. Jarvis, and G. Winstanley. Reducing the Semantic Gap Between Application Domain and AI Planning Technology: a compilation approach *In: Proceedings of The Workshop on Knowledge Acquisition and Knowledge Elicitation*, held during the Fourth International Conference on Artificial Intelligence Planning Systems, Pittsburgh, USA. Available as AAAI Technical Report WS-98-03. Available from *http://www.aiai.ed.ac.uk/~paj /paj-pubs.html*

[Jarvis & Winstanley 1996a] P. Jarvis, and G. Winstanley. Dynamically Assessed and Reasoned Task (DART) Networks. *In: Proceedings of the 16th International Conference of the British computer Society Specialist Group on Expert Systems*, Cambridge, UK.

[Jarvis & Winstanley 1996b] P. Jarvis, and G. Winstanley. Objects and Objectives: the merging of object and planning technologies. *In: Proceedings of the 15th Workshop of the UK Planning and Scheduling Special Interest Group*, Liverpool, UK.

[Jarvis 1997] P. Jarvis. Integration of Classical and Model-Based Planning. *Phd Thesis, School of Computing and Mathematical Sciences, The University of Brighton*, Sussex, UK. *available from: http://www.aiai.ed.ac.uk/~paj/thesis/*

[Kambhampati 1994] S. Kambhampati. Comparing Partial Order Planning and Task Reduction Planning: A Preliminary Report. *In: Working Notes Of the Workshop on the Comparative Analysis of Planning Systems*, held during the 12th National Conference on Artificial Intelligence (AAAI-94), Seattle, USA.

[Schreiber et al. 1993a] G. Schreiber, B. Wielinga, and J. Breuker, eds. *KADS: A Principled Approach to Knowledge-Based System Development.* Academic Press.

[Schreiber et al. 1993b] G. Schreiber, B. Wielinga, and J. Breuker. Introduction and Overview. *In: [Schreiber et al. 1993a]*, 6-7.

[Tate et al. 1994a] A. Tate, B. Drabble, and J. Dalton. O-Plan Version 2.2 Task Formalism Manual. O-Plan Project Documentation, AIAI, The University of Edinburgh, 80 South Bridge, Edinburgh, UK. *available from: http://www.aiai.ed.ac.uk/~oplan/oplan/oplan-doc.html*

[Tate et al. 1994b] A. Tate, B. Drabble, and J. Dalton. The Use of Condition Types to Restrict Search in an AI Planner. *In: Proceedings of the 12th National Conference on Artificial Intelligence (AAAI-94)*, Seattle, USA. *available from: http://www.aiai.ed.ac.uk/~oplan/oplan/oplan-doc.html*

[Tate et al. 1998] A. Tate, S. Polyak, and P. Jarvis. TF Method: An Initial Framework for Modelling and Analysing Planning Domains. *In: Proceedings of The Workshop on Knowledge Acquisition and Knowledge Elicitation*, held during the Fourth International Conference on Artificial Intelligence

Planning Systems, Pittsburgh, USA. Available as AAAI Technical Report WS-98-03. Available from *http://www.aiai.ed.ac.uk/ paj/*

[Tate 1977] A. Tate. Generating Project Networks. *In: Proceedings of the International Joint Conference on Artificial Intelligence*, 888-893.

[Tate 1996] A. Tate. Representing Plans as a Set of constraints - the <I-N-OVA> Model. *In:* B. Drabble, ed., *Proceedings of the Third International conference on Artificial Intelligence Planning Systems (AIPS-96)*, Edinburgh, UK, May 1996, AAAI Press. *available from: http://www.aiai.ed.ac.uk/~oplan/oplan/oplan-doc.html*

[Wilkins 1988] D. Wilkins. *Practical Planning: Extending the Classical AI Planning Paradigm.* Morgan Kaufmann, USA.

[Wilkins 1997] D. Wilkins. Using the SIPE-2 Planning System: A Manual for Version 4-17. SRI International Artificial Intelligence Center, Menlo Park, CA, USA. *available from: http://www.ai.sri.com/~wilkins/bib.html*

[Winstanley & Hoshi] G. Winstanley, and K. Hoshi.. Activity Aggregation in Model-Based AI Planning Systems. *International Journal of Artificial Intelligence in Engineering Design, Analysis and Manufacture (AI EDAM)*, 7(3), 209-228.

[Yang 1990] Q. Yang. Formalising Planning Knowledge for Hierarchical Planning. *Computational intelligence*, 6(1), 12-24.