# INTERMEDIATE FUNCTION ANALYSIS FOR IMPROVING CONSTRUCTABILITY

**SONG YUANBIN**

**(B. Eng, Southeast University)**

**(M. Eng, Southeast University)**

**A THESIS SUBMITTED**
**FOR THE DEGREE OF DOCTOR OF PHILOSOPHY**
**DEPARTMENT OF CIVIL ENGINEERING**
**NATIONAL UNIVERSITY OF SINGAPORE**

**2006**

# ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor, Associate Professor David Kim Huat Chua, for his advice, support, patience, and encouragement throughout the course of this research. It is not often that one finds an advisor who is always energetic and active in both academic and consultancy fields. Particularly, it is his good relationships with the local industries that make it possible for me to access the real project management and project cases and to verify the research results in the real project management environment. His advice was essential to the completion of this dissertation as well as many academic papers.

I convey my sincere appreciation to Associate Professor Chan Weng Tat, for his constructive and valuable suggestions in the initial stage of the present research. My sincere appreciation also goes to the members of the research committee, Associate Professor Choo Yoo Sang and Associate Professor A. Senthil Kumar for providing many valuable comments in the qualification examination. My sincere appreciation is also extended to Professor Cheng Hu in Southeast University. His insightful commentaries and opinions on many research topics are very helpful for this work. Moreover, I also appreciate the friendship and kindness of Professor Cheng in the past decade.

I would like to express my special thanks to Mr. Yeoh Ker-Wei, who spent a lot of time, energy, and patience to help proofread the thesis draft. I also enjoyed the times when we discussed many research topics on spatio-temporal analysis. I am also grateful to my classmate Ms. Chen Qian, who provided several valuable suggestions for the second case study.

I am grateful to Mr. Kuo Li Ho, senior project manager of Bored Piling Pte. Ltd., Dr. Daniel Lim, vice president of SembCorp Engineers & Contractors Pte. Ltd.

and Mr. Peter C.Y. Ho, Head of Construction Department of JGC Singapore Pte. Ltd., for their extensive support in the interviews and the case studies. I also express my thanks to other staff in the above-mentioned companies for their generous provision of time and knowledge in site visits and interviews.

My thanks also go to the management staff of the Traffic Laboratory and Educational and Information Technology Laboratory for their technical support. My appreciation is also given to the undergraduate students who helped in drawing the 3D models.

My genuine acknowledgement is given to National University of Singapore for providing research scholarship and to Infocomm Development Authority (IDA) of Singapore for sponsoring the Collaborative Engineering Program (CEP) research project with SGD 1.4M for 3 years.

Last, but not least, I would like to thank my wife Lily for her love and care during the past three years. Her understanding and encouragement was in the end what made this dissertation possible. I am indebted to her for the lonely days for taking care of my daughter as well as my parents alone. I also thank my cute daughter Sarah, who frequently used her shining smile to entertain me through the tough journey of writing. My parents and my grandmother should also deserve my deepest gratitude and love for their dedication, encouragement and support, which paved the way to this work.

# TABLE OF CONTENTS

# SUMMARY

The Architecture/Engineering/Construction (AEC) industry still lacks an approach to represent and analyze intermediate function requirements arising from supporting the construction processes and maintaining the temporary stability of in-progress structures. This inadequacy may greatly affect the capability of constructability analysis with respect to the executability of construction schedules. Thus, the present research attempts to develop an approach to represent and analyze intermediate function requirements.

The component state concept and In-Progress Product Core Model (IPPCM) as well as Product-Oriented Scheduling Technology are developed to abstract the in-progress configuration of a facility product using a component state network. Each component state has both temporal and spatial attributes. In this way, the construction life cycle of a product component can be described in terms of a state chain, and the functional dependencies between two in-progress product components can be abstracted with respect to interval-to-interval relationships between component states. Furthermore, the duration of a component state is further divided into an active phase and a quiescent phase, leading to better description of the requirement and availability conditions of intermediate functions.

An intermediate function can be semantically modeled in five layers. Based on such a semantic model, intermediate function requirements can be evaluated from both temporal and spatial perspectives. Moreover, the temporal logics residing in construction methods can be captured as intermediate function requirement knowledge from three perspectives, namely the construction life cycle of a single component, the functional interdependencies between two in-progress components, and the availability condition of an intermediate functionality with respect to a group

of in-progress components. A schema for representing this knowledge has been developed using two product-oriented constructs, namely component type and state type, and four categories of temporal interval relationships, which are precedent, coincident, coupling, and disjoint relationships.

An information framework is developed for intermediate function analysis. This framework integrates five project modeling perspectives, namely product, process, intermediate function, space, and resource. Based on such a framework, four analysis methodologies have been developed. The first and second analysis methodologies can be used for detecting unfulfilled intermediate function requirements from the temporal and spatial perspectives, respectively. The third analysis method facilitates resolving compatible intermediate function requirements by co-matching multiple users and providers from different trades, and the fourth method can be applied for identifying bottleneck states which determines the earliest availability of intermediate functionalities.

A software prototype 4D Intermediate Function AnalysiS Tool (4D-iFAST) is developed for implementing the information integration framework and the analysis methodologies as well as 4D simulation. Additionally, the existence vector together with the Boolean operations simplifies the time-window analysis for intermediate function analysis, and also makes it possible to implement spatio-temporal analysis without having to conduct 4D simulation. Two industry cases are used for validating the developed intermediate function analysis tools. These case studies indicate that the construction period can be shortened and that the collaboration on realizing intermediate functions among trades can be improved by using the developed tools.

# NOMENCLATURE

| | |
|---|---|
| IPPCM | In-Progress Product Core Model |
| AEC | Architecture/Engineering/Construction |
| POST | Product Oriented Scheduling Technique |
| 4D-iFAST | 4D intermediate Function AnalysiS Tool |
| DFM | Design For Manufacturability |
| GIS | Geographic Information System |
| EPM | Extended Product Model |
| STP | State Transition Point |
| CPM | Critical Path Method |
| S | Start (time point attribute of state) |
| AF | Active (time point attribute of state) |
| F | Finish (point attribute) |
| SD | State Duration (interval attribute) |
| AD | Active Duration (interval attribute) |
| QD | Quiescent Duration (interval attribute) |
| PS | Performer State |
| .A | Active Phase of State |
| .Q | Quiescent Phase of State |
| $C_i.S_j$ | The $S_j$ state of the $C_i$ component |
| $I(C_i.S_j)$ | The duration interval of the component state $C_i.S_j$ |
| $C_i.S_j.A$ | The active phase of the component state $C_i.S_j$ |
| $C_i.S_j.Q$ | The quiescent phase of the component state $C_i.S_j$ |
| RSP(F) | The requirement state package of the intermediate function F |

| | |
|---|---|
| FSP(F) | The functional state package of the intermediate function F |
| RTW(F) | Requirement Time-Window of the intermediate function F |
| ATW(F) | Availability Time-Window of the intermediate function F |
| NMTW(F) | Non-Matching Time-Window between the user and the provider of the intermediate function F |
| mNMTW($F_1$, …, $F_n$) | Non-Matching Time-Window between the users and the providers of *n* intermediate functions $F_1$ to $F_n$ |
| $\cup$ | Boolean Union |
| $\cap$ | Boolean Intersection |
| - | Boolean Cut |
| $SE_m$ | An arbitrary space entity in the space model |
| $TSE_m$ | An arbitrary temporal space entity in the space model |
| EP($TSE_m$) | The existence period of the temporal space entity $TSE_m$ |
| TTR($TSE_x$, $TSE_y$) | The temporal topological relationship between the temporal space entities $TSE_x$ and $TSE_y$ |
| EP(TTR($TSE_x$, $TSE_y$) | The existence period of the temporal topological relationship between $TSE_x$ and $TSE_y$ |
| M | Meet |
| I | Intersect |
| D | Disjoint |
| =EP(Use) | The existence period of the temporal topological relationships should be equal to the associated user space entity. |
| $EV_a$($f_1$, …, $f_i$, …, $f_n$) | An arbitrary existence vectors whose sizes are n, and $f_i$ is the $i^{th}$ element of $EV_a$ (at the $i^{th}$ finite interval unit) |

# LIST OF FIGURES

# LIST OF TABLES

# CHAPTER 1 INTRODUCTION

## 1.1 Research Motivation and Background

This dissertation presents the main works of the research project In-Progress Product Core Model (IPPCM), a subproject of the Collaborative Engineering Program (CEP), which is a collaboration sponsored by the Infocomm Development Authority of Singapore (IDA), National University of Singapore (NUS), Sun Microsystems, the Asia Pacific Science & Technology Center (APSTC) of Sun, and Singapore Technologies Electronics (Info-Software Systems) Pte. Ltd. One critical motivation of the IPPCM project is to help the Architecture/Engineering/Construction (AEC) industry improve the constructability of a facility project through the systematic analysis of construction requirements. In this connection, the construction requirements should be represented, communicated, and then evaluated before the commencement of project construction.

Construction requirements are capabilities and conditions to which both the construction process system and the in-progress facility product must conform. Otherwise, the construction processes may be delayed or the temporary stability of the in-progress structure may not be sustained during construction. Similar to software requirements (Cysneiros and Yu 2004), construction requirements can be classified into two categories: functional and non-functional. Functional construction requirement defines the temporary functionalities required by in-progress facility products and construction performers, while non-functional requirement indicates the availability and performance capacity of construction resources. The fulfillment of the former generally requires the support of the in-progress facility, while the fulfillment of the latter indicates the availability of the construction resources that are prerequisites for construction processes. Specifically, the constraint-based scheduling

research (Shen and Chua, 2005; Chua and Shen, 2005; Chua et. al., 2003) focuses on incorporating non-functional construction requirements into construction schedules.

The functional construction requirements can be further divided into two subcategories: transformation functions and intermediate functions. Transformation function describes different types of operational functionalities required for transforming the material compositions, shapes, and locations of product components or resource components, while intermediate function represents various kinds of functionalities provisionally required for supporting the construction performers and for maintaining the temporary stability of an in-progress structure. The present study focuses on analyzing intermediate functions.

Additionally, more types of intermediate functionalities will be discussed in Section 4.1 of Chapter 4. Besides supporting construction loads and maintaining stability of in-progress structures Intermediate functions, intermediate functions are also required for providing a workface, providing protection for finished works and providing safe work environments. This research will concentrate on analyzing the first two subcategories of intermediate functionalities. This analysis may help designers and constructors to identify the unfulfilled intermediate function requirements and then resolve them to improve the constructability of a facility project.

## 1.2 Construction Requirement Analysis for Improving Constructability

The AEC trades have recognized that systematic analysis of construction requirements, especially intermediate function requirements, plays an indispensable role in improving the constructability of a facility project. More and more clients are

keeping prudent watch on the cost of realizing the intermediate functions with respect to the selection of construction methods. Designers are becoming aware of the importance to concurrently consider both usage requirements from the clients and the intermediate function requirements from the constructors, while the specialist constructors and fabricators should make their special intermediate function requirements known to the designers as early as possible. The construction contractors and subcontractors should also collaboratively plan their construction schedules to ensure that the upstream works can provide the intermediate functionalities for executing the downstream processes. Meanwhile, construction schedules should also be examined from the intermediate function viewpoint in order to ensure the accessibility of labor and heavy equipment and to make certain the temporary stability of the in-progress structure as well as to reduce interferences between/among trades. Furthermore, several alternatives for resolving the intermediate function requirements may be explored in order to shorten construction schedules and decrease excessive expenditure on temporary facilities.

From a pragmatic viewpoint, early consideration and evaluation of crucial intermediate function requirements can result in improved executability of a construction schedule, which is a key aspect of constructability. The improved executability often benefits the constructors in higher productivity and safer work environment, leading to profit increase. Meanwhile, the improved executability of a construction schedule can also benefit the designers by decreasing the number of change orders arising from the late identified intermediate function requirements, resulting in earlier delivery of engineering drawings with improved constructability.

## 1.3 Challenges for Intermediate Function Analysis

Although the AEC industry has been aware of the importance of constructability analysis for decades and even developed various programs to improve constructability, the systematic analysis of intermediate function requirements is still limited. A major reason is that the AEC practitioners still encounter at least four challenges in analyzing intermediate function requirements. The inadequate evaluation frequently results in project delays and additional costs in the form of frequent change orders, increased reworks, low productivity, and work space congestion as well as expensive and unsafe access to the in-progress works.

Firstly, the AEC industry still lacks a semantic model to represent the intermediate function requirements. Such requirements are frequently represented in the format of natural language. Sometimes, an even worse situation is that the intermediate function requirements and the knowledge to resolve these requirements are only stored in the engineers' mind instead of being recorded on paper or in computer systems. The natural-language-based representation may cause ambiguity among the participants, and also makes it very difficult for employing information technology to facilitate the analysis of intermediate function requirements. Moreover, this may also hinder the communication of intermediate function requirements among the participant trades, especially those dispersed in distinct engineering fields.

Secondly, the current integration among prevailing project management software is inadequate for rendering the information required for analyzing the intermediate function requirements. The integration between the construction requirement modeling perspective and the other project modeling perspectives, like product, process, resource, and space, is still rudimentary and unstructured. This means that the AEC practitioners lack an information integration framework for

conducting intermediate function requirement analysis. Additionally, the delay of constructability improvement ideas may also be exacerbated due to lack of an information integration framework.

Thirdly, the AEC industry still lacks methodologies to analyze the intermediate function requirements. Although many AEC companies have established internal constructability improvement programs and constructability review procedures, the analysis of intermediate function requirements is frequently conducted ad hoc instead of in a systematic manner, leading to construtability improvement decisions that are too late to be applied. A major reason is that construction engineers lack analysis tools for systematic analysis of intermediate function requirements.

Lastly, the inefficient practice of intermediate function analysis may also arise from the fragmented nature of facility project management. Angelides (1999) has classified the fragmentation of project management into three categories, namely, sequential realization, segmented view of product quality, and fragmented project control. Specifically, some research indicates that such project perspectives as construction scheduling and cost estimating are often managed and optimized from the viewpoint of a specific organization rather than from an overall project perspective (Hendrickson and Au, 1989).

Another fragmentation category is the different modeling perspectives employed by different trades for managing construction requirements. For example, designers tend to evaluate and specify construction requirements from the product perspectives, while constructors often specify their construction requirements with respect to construction schedules. This often results in that the solutions for resolving some intermediate functions, when optimized only within one organization, may impair overall constructability. Specifically, some trades may be unaware or negligent

of their responsibilities for realizing the intermediate functionalities required by the fellow trades.

A comprehensive literature review in the following chapter shows that construction requirement management has been studied along several research trends. These studies have made significant contribution to improve the constructability of facility projects. However, the AEC project management community still finds it difficult to derive an analysis framework or approach from these previous studies in order to represent and evaluate intermediate function requirements. Specifically, the requirement and availability of an intermediate functionality is inadequately studied in many previous studies, while the 4D research does not provide adequate information for evaluating the time-dependent spatial interaction between the users and the provider of an intermediate functionality.

## 1.4 Research Objectives

This research project primarily attempts to develop a framework for intermediate function analysis. Such a framework will comprise the concept and semantic model for representing the intermediate function, the representation schema for describing the intermediate function requirement knowledge, the information integration framework for deriving the attributes of intermediate functions, and the analysis methodologies for detecting unfulfilled intermediate function requirements. The present study also attempts to explore the feasibility of using 4D simulation to facilitate intermediate function analysis. In this way, the executability of a construction schedule can be improved, consequently leading to improved constructability of a facility project and better collaboration among the trades.

To achieve this general goal, this research project is intended more specifically for delivering the following research components:

**(1) Model to Describe In-progress Configurations of Facility**

This research seeks to extend the traditional product decomposition model for describing the configuration of an in-progress facility and for representing intermediate functions. Accordingly, a scheduling method will be developed to derive temporal attributes associated with the in-progress facility product.

**(2) Concept and Semantic Model to Abstract Intermediate Functions**

The present research attempts to develop the concept and semantic model to abstract an intermediate function requirement. Such a concept should be less dependent on a specific engineering domain so that it can be easily understood and applied by the trades distributed in different engineering domains. Accordingly, the semantic model should allow integrating the intermediate function modeling perspective with other project modeling perspectives like product, process, and space perspectives. In this way, the temporal and spatial attributes in other models can be mapped onto the intermediate function model.

**(3) Schema for Representing Intermediate Function Requirement Knowledge**

A schema for representing intermediate function requirement knowledge will be developed for capturing the temporal logics residing in construction methods, especially those concurrent relationships. Such a knowledge representation schema can also be used for facilitating the description of in-progress facilities and the analysis of intermediate function requirements.

**(4) Information Integration Framework**

An information integration framework will be developed for associating the intermediate function modeling perspective with such project modeling perspective models as process, product, resource, and space. These modeling perspectives are required for deriving temporal and spatial attributes for intermediate function analysis.

**(5) Intermediate Function Analysis Methodologies**

The present study attempts to develop analysis methodologies for evaluating the temporal and spatial perspectives of intermediate functions requirements, since these two perspectives are the common characteristics of all intermediate function requirements. These analysis methodologies can be used for detecting the unfulfilled intermediate function requirements. Meanwhile, this study also plans to develop an analysis methodology for identifying the critical factors that determine the availability of some intermediate functionalities, which restrict the commencement of the associated construction activities. This may help planning engineers reduce construction periods.

**(6) Software Prototype for Implementing Analysis Methodologies**

A software prototype will be developed to implement the information integration framework and the analysis methodologies as well as 4D simulation. Additionally, the capability of the 4D simulation for facilitating intermediate function analysis will also be explored using the prototype.

**(7) Case studies for Validating the Developed Analysis Framework**

This research will validate the developed intermediate function analysis framework with two case studies. The first is the construction of a bridge deck using balanced cantilever approach, while the second is the construction of the entrance gate of a nursing home. These two case studies will be intentionally amended to keep confidential some sensitive data, while the characteristics of the evaluated intermediate function requirements should be kept as original.

## 1.5 Research Methodology

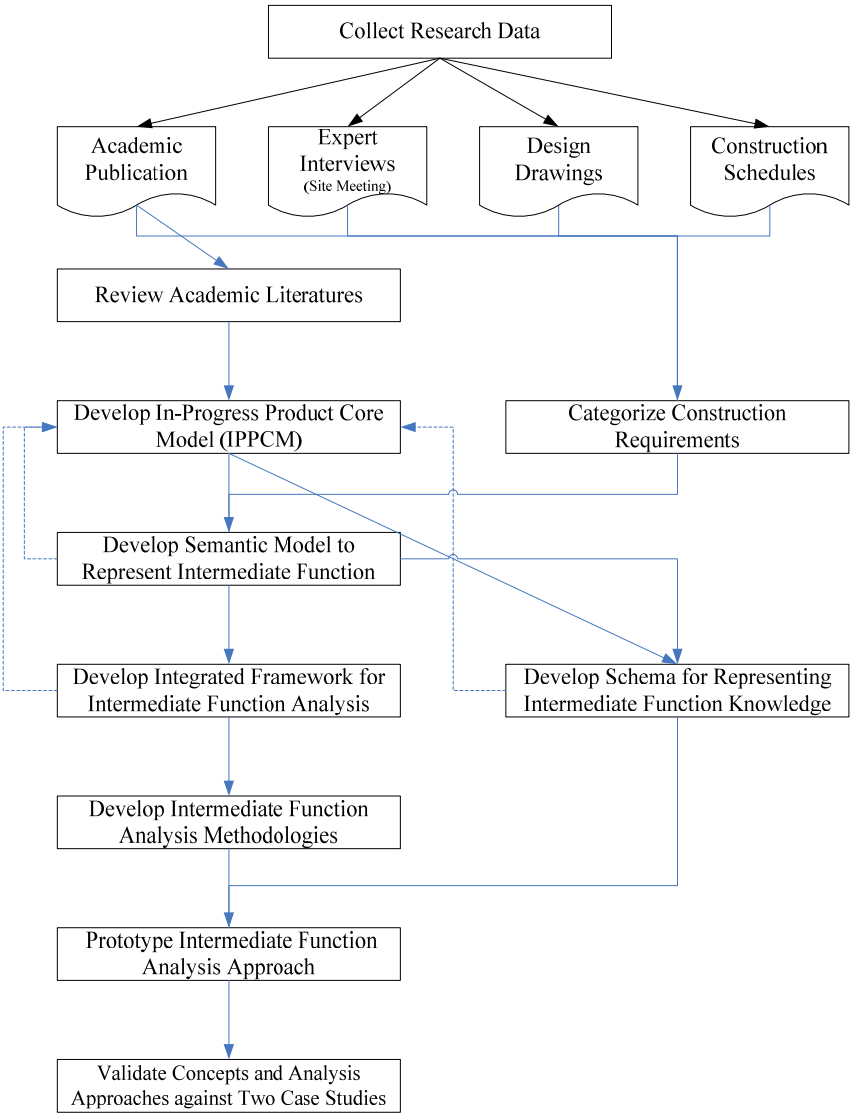



**Figure 1.1 Research Procedure**

The present research project adopts the research methodology illustrated in Figure 1.1. The Figure shows the sequence of the key research steps, which are explained as follows:

**(1) Collect Research Data**

In the initial stage, various types of research data related to construction requirement management were collected for the succeeding works. There were four types of data: academic publication, expert interviews, design drawings, and construction schedules. There were more than 300 papers collected and reviewed, and around two thirds of them were referenced by the present research.

The author of the present research had attended site meeting for more than 100 hours in order to understand the current practice of construction requirement management as well as to collect various construction requirements. Meanwhile, the author also conducted face-to-face interviews with 24 AEC experts. Among them, there were one directing manager of a construction company, two senior project managers, twelve construction site engineers, three construction planner, three designers, and three consultants for project management. These interviews were required for understanding the barriers in construction requirement management and also for collecting the suggestions on improving the current construction requirement management. These interviews also helped verify the developed intermediate function analysis framework.

Two real cases had been collected for the present research with respect to the design drawings and the construction schedules as well as other project documents like site photos and progress records. The two case studies were intentionally amended to keep confidential some sensitive data, while the characteristics of the evaluated intermediate function requirements were kept as original.

**(2) Review Academic Literatures**

The review of academic literatures covers such research fields as constructability analysis, function modeling, facility product modeling, construction sequence modeling, concurrent relationship modeling, and space requirement analysis as well as 4D simulation. This review provided a solid foundation for the present research, and also helped explore the inadequacy of the developed function modeling approaches and integrated information frameworks for construction requirement analysis.

**(3) Categorize Construction Requirements**

The construction requirements gathered through case studies and site interviews as well as academic literatures were categorized by capturing their key characteristics or attributes since the AEC industry has not proposed a categorization schema or terminology dictionary for classifying construction requirements. Such semantic technologies as ontology modeling approach may be helpful in this research stage. The preliminary study indicated that requirements can be divided to functional and non-functional sets, which require different representation models and analysis approaches. Therefore, construction requirements were categorized at early stage in order to scope the research. Particularly, intermediate function requirement is the focus of the present study.

**(4) Develop In-Progress Product Core Model (IPPCM)**

The In-Progress Product Core Model (IPPCM) was developed before addressing the concept of intermediate function since IPPCM comprises the "component state" concept that is necessary for the semantic representation of an intermediate function. The temporal and spatial attributes of a component state will be defined. Particularly, the temporal relationships between component states will be

represented using interval-to-interval relationships developed in the interval algebra instead of traditional point-to-point precedent relationships.

**(5) Develop Semantic Model to Represent Intermediate Functions**

The semantic representation of intermediate function requirement knowledge is explored in this research step. Literature reviews have indicated that there are several ways to model a general function. According to the categorization of construction requirements, this study concentrated on developing the semantic model of intermediate function, which is a subcategory of functional construction requirements. The present study defined an intermediate function from both temporal and spatial perspectives, assuming that an intermediate function can be represented as the temporal and spatial interaction between the user and the provider.

**(6) Develop Schema for Representing Intermediate Function Requirement Knowledge**

Based on the developed IPPCM and the semantic model of intermediate functions, the temporal logics residing in construction methods can be captured in terms of intermediate function requirement knowledge. Accordingly, the representation schema can be developed. Meanwhile, the study in this stage also helps improve the concepts and methodologies for modeling an in-progress product.

**(7) Develop Integrated Framework for Intermediate Function Analysis**

The developed IPPCM provides a kernel to integrate the product, process, and space perspectives of a facility project. Such an integration structure should be further extended to integrate the resource model and the intermediate function model, since either a function provider or a function user may comprise resource components like labor and heavy equipment. In this way, the integration framework, linking five

project modeling perspectives, can provide the information for analyzing intermediate function requirements using the analysis tools developed in the next step.

**(8) Develop Intermediate Function Analysis Methodologies**

The intermediate function analysis can be conducted from three perspectives, namely temporal interaction between user and provider, spatial interaction between them, and bottleneck state for determining the availability of an intermediate functionality. Accordingly, the intermediate function analysis tools can be developed from these three viewpoints. The analysis results can be used to detect the unfulfilled intermediate function requirements. It was also expected that the analysis results can help elicit alternative solutions to better realize the intermediate function requirements.

**(9) Prototype Intermediate Function Analysis Approach**

A software prototype will be developed for two main purposes: to implement the developed analysis approaches and to conduct 4D simulation for visualizing IPPCM. Accordingly, the two key parts of the prototype are the inference engine to evaluate the temporal and spatial interactions and the 4D simulation engine for visualizing the construction progress described by the IPPCM. The Access desktop database will be used for storing the model data, and the Delphi 5 (Object Pascal language) will be used for coding most parts of the prototype software. Additionally, the prototype will import the construction schedules generated by Ms Project 2003 and the 3D models created by AutoCAD 2000.

**(10) Validate Concepts and Analysis Methodologies against Two Case Studies**

The developed concepts and analysis methodologies will be validated against two case studies collected from industry experts. Moreover, the engineering meaning or explanation of the inference results will be further studied in this research stage.

Additionally, Figure 1.1 also shows that some research results produced in the downstream stages were incorporated into the earlier stages for enhancing the corresponding research results. Specifically, the development of the semantic model for representing intermediate functions required that the component state concept developed in the preceding step be amended by including the spatial attribute in order to accommodate the representation and evaluation of the spatial perspective of an intermediate function. Meanwhile, the development of the semantic model of intermediate functions indicated that the distinction of active and quiescent phases of a component state can better describe the requirement and availability of an intermediate function, and the study of cross-component state relationships also indicated that this distinction can facilitate representing functional interdependencies between in-progress product components.

## 1.6 Organization of Dissertation

This dissertation is organized as follows:

This chapter introduces the background and motivation of the present research. The importance of conducting intermediate function requirement analysis for improving the constructability of a facility project is addressed, and the challenges arising from intermediate function analysis are also outlined. In this connection, the research objectives and research methodology are stated. Additionally, the organization of this thesis is also described by summarizing the contents of each chapter.

Chapter 2 presents a comprehensive review of the academic publications related to the present research. The literature reviews on constructability analysis indicate that construction requirement analysis should be the fundamental issue of

constructability analysis. Subsequently, such research topics as modeling engineering requirements, modeling facility product, representation of construction sequencing requirements, concurrent relationships among activities, and modeling space requirements are respectively reviewed. These reviews indicate that a systematic approach for analyzing intermediate function requirement should be developed to advance the research and practice of construction requirement analysis.

Chapter 3 presents the key issues for developing an In-Progress Product Core Model (IPPCM). The structure of the IPPCM is presented, and this is followed by introducing the concept of component state along with its temporal and spatial attributes. Based on this concept, the construction life cycle of a product component can be represented as a sequence of component states, and the functional interdependencies between two in-progress components can be described with respect to interval-to-interval state relationships. In this way, the in-progress configuration of a facility product can be described by a component state network. Lastly, the Product Oriented Scheduling Technique (POST) is developed for deriving the temporal attributes of component states.

Chapter 4 provides a semantic model for representing an intermediate function in five layers. Subsequently, this chapter presents a schema for representing intermediate function requirement knowledge. It is developed for capturing the temporal logics residing in construction methods from three perspectives, namely the construction life cycle of a single product component, the functional interdependencies between two in-progress components, and the availability conditions for a group of provider components. This chapter also presents an information integration framework for integrating five project modeling perspectives required by intermediate function analysis. These five perspectives are product,

process, resource, space, and intermediate function. The integration is realized through work package, performer state package, requirement state package, and functional state package as well workspace and state space.

Chapter 5 presents four analysis methodologies. The first and second methodologies are developed for evaluating the temporal and spatial interactions between the user and the provider of a single intermediate function. The third methodology extends the first methodology from a single intermediate function to a set of compatible intermediate functions by co-matching their requirement and availability time-windows. The fourth methodology can be used for identifying bottleneck states that constrain the commencement of the associated construction activities.

Chapter 6 presents the research prototype 4D intermediate Function AnalysiS Tool (4D-iFAST) to implement the information integration framework and the analysis methodologies as well as 4D simulation. The potential benefits from using 4D simulation for intermediate function analysis is first explored, and then the architecture and the main data structure of the prototype are presented. The mechanisms of the inference engine for detecting unfulfilled requirements and the 4D simulation engine for visualizing construction schedule are subsequently introduced. Particularly, the existence vector concept and the corresponding Boolean operations are developed, which make it feasible to detect the unfulfilled temporal and spatial interactions without the need to conduct simulation.

Chapter 7 presents two case studies used for validating the developed intermediate function analysis framework. These two case studies also demonstrate the application of the developed concepts, semantic model, knowledge representation schema, and analysis methodologies. The first case study illustrates the representation

of intermediate function requirement knowledge and also demonstrates the identification and analysis of the bottleneck states, while the second case study is used to illustrate the co-matching between the requirement time-windows and availability time-windows of two substitutable temporary support functions.

Chapter 8 first summarizes and discusses the research findings and then presents the limitations and the future works.

Finally, the Appendix contains a list of publications in journals and conferences arising from the current research that have been published or accepted for publication up to this point in time.

# CHAPTER 2 LITERATURE REVIEW

The present study relates to such research fields as constructability analysis, function modeling, facility product modeling, representation of construction sequences, concurrent relationships, and space requirements in construction planning as well as 4D simulation. These research findings indicates that the previous studies are inadequate for analyzing intermediate function requirements.

## 2.1 Construction Requirement Analysis for Improving Constructability

The concurrent engineering philosophy advocates that the requirements occurring in the downstream activities should be considered and evaluated in the upstream activities as early as possible. A number of studies indicated that Design For Manufacturability (DFM) is an important approach to implement concurrent engineering philosophy in developing manufacturing products (Sapuan et. al., 2006; Pham and Dimov, 1998; Braunsperger, 1996; Youssef, 1994; Yeh, 1992; Ranky, 1994). Compared with the traditional approaches for managing the development life cycle of a manufacturing product, the DFM idea stresses that the manufacturing/processing requirements should be considered and evaluated during the design phase. Besides fabrication and assembly requirements, the manufacturing requirements may cover the requirements from procurement, shipping, test, and even maintenance. In a DFM team, these manufacturing requirements from the factories are often concurrently considered with the usage requirements from the customers. In this way, many potential conflicts arising from manufacturing processes can be identified and proactively prevented, leading to high productivity and reliability with decreased defective product ratio.

Similar to the application of DFM in the manufacturing industry, the AEC industry has mooted its constructability concept as a critical approach for improving the development of a facility product. The constructability concept stresses the optimum usage of construction knowledge in the early development phases such as design and planning in order to achieve overall project objectives. (CII, 1986; Construction Management Committee of the ASCE Construction Division, 1991). This approach highlights the importance of managing construction requirements since a large part of construction knowledge and experience can be represented and managed with respect to construction requirements. Therefore, construction requirements should be one of the fundamental and dominant concepts in constructability analysis.

The constructability concept further stresses that the construction requirement analysis should be conducted as early as possible. Specifically, the construction requirements should be incorporated into both facility designs and construction schedules in early project phases (CII, 1986; CII, 1987[a]). The less effective “late constructability review” program should be replaced with the more efficient “early constructability analysis” approach (CII, 1987[a]). The “”late constructability review” program is often conducted when design has been completed or nearly finished. At that stage, even if some valuable solutions to resolve the crucial construction requirements have been produced, these solutions may not be incorporated into the related designs and schedules due to tight project schedule and the potential adverse relationships among the related parties. This means that the construction requirements should be identified in the early development phases to make sure that some critical construction requirements may not be omitted or neglected by the project players.

The importance of managing construction requirements can also be seen from the guidelines to improve constructability in such development phases as conceptual planning (CII, 1987[b]; Tatum, 1989), engineering and procurement (CII, 1986[c]; O'Connor et. al., 1987), and even field operations (CII, 1988; O'Connor and Davis, 1988; Fisher and O'Connor, 1991). These principles for eliciting constructability improvement ideas suggest that the various construction requirements should be systematically identified and incorporated into the analysis during pre-construction stages. Specifically, project plans, designs, procurement schedules, and site layouts should be 'construction-driven'.

Several previous studies indicate that managing construction requirement information plays an indispensable role in a successful constructability improvement program. A number of constructability improvement programs have been proposed specifically for managing construction requirement information from different management scopes and levels (Russell and Gugel, 1993; Radtke and Russell; 1993; Russell et. al., 1994). A common focus of these programs is to manage various channels and interfaces for exchanging the construction requirement information and for incorporating this information into the facility designs and construction plans.

Some studies indicate that the lack and delay of construction requirement information is a major barrier causing a hindrance to the improvement of constructability (CII, 1993[c]). Specifically, the timely exchange of construction requirement information between the workflow of constructability review processes and the workflow of project development processes should be regarded as one of the crucial factors of successful constructability improvement analysis (Anderson et al. 2000). This means that if the construction requirement information is not

systematically recorded, communicated, and analyzed, the constructability of a facility project may be compromised.

## 2.2 Engineering Function Modeling and Analysis

In general, a lot of design intentions and rationale can be represented by functions. Function modeling is an approach to elicit, express, and evaluate the design intentions of an artifact (Lee, 1997). Functionality modeling can be used for reasoning the design rationale of mechanical and electrical products (Chakrabarti and Blessing, 1996; Chakrabarti and Bligh, 1996; Deng et. al., 1998; Deng, 2000, Deng et,. al., 2000[a]; Deng et,. al., 2000[b]; Deng et,. al., 2000[c]). Function modeling can also be applied for redesigning a product, whereby the designers can locate functional redundancies by exploiting potential functionalities of the designed components (Umeda et; al., 1992). Additionally, other researchers have explored the application of function modeling in fault diagnosis (Hawkins and Woollons, 1998; Kumar and Upadhyaya, 1995; Hawkins, 1994). This implies that intermediate function requirement can be also be modeled from the function perspectives.

Moreover, some studies indicate that the function model of the artifact is necessary to convey the engineering behaviors or functionality characteristics that cannot be represented solely by its product model. This is because the mapping relationship between the functionality elements and the product components is many-to-many instead of one-to-one (Kumar and Upadhyaya 1995).

Some ontological studies indicate that function model can be used to explicitly communicate designer's intentions across different technical domains (Kitamura and Mizoguchi, 1999; Kitamura and Mizoguchi, 2003). Specifically, Sasajima et. al. (1995) developed a function knowledge representation schema that comprises a set of

generic functions and a collection of meta-functions containing the interdependencies between functions. This schema facilitates explaining the design intentions of a complex engineering system across a design team, whose members may come from different engineering domains.

The function concept plays an important role in representing and communicating requirements and intentions in various engineering domains, but the term “function” seems too overloaded in different contexts. In many previous studies, the term “function” is frequently confused with the terms “behavior” or “purpose”. Several studies suggested that “function” should be clearly distinguished from “behavior” and “purpose” (Chittaro et; al., 1993; Larsson, 1996). The term “purpose” is described from the perspective of the user, who can select different engineering solutions to achieve it. On the other hand, the “behavior” is inherent to the product (engineering solution), and is immaterial of the purposes of the potential user. This distinction can help explicitly represent functional purposes and the engineering solutions to achieve them.

Previous studies indicate that a function can be represented from three viewpoints: user, provider/product, and the relationship between them. Chittaro and Kumar (1998) suggested that a function can be defined from two perspectives: operational and purposive. The former is a product-oriented definition that a function is a relationship between the input and output of a product, while the latter is a user-oriented definition that depicts a function as the purpose, goal, or intention from users of a specific product. Similarly, a function is also suggested to be modeled from either purpose (user-oriented) or action (provider-oriented) perspective (Winsor and MacCallum, 1994). Likewise, Chandrasekaran and Josephson (2000) suggest that a function can be described from either an environment-oriented viewpoint (similar to

purpose) or a device-oriented viewpoint (similar to operation/action). Chakrabarti (1998) proposes that a function can be abstracted as behaviors in the component level (provider-oriented) and purposes at a higher structure/system level (user-oriented).

On the other hand, there are also other studies that define function as the relationships between the users' requirements/purposes and the engineering behaviors exhibited by a product system (Bobrow, 1984). The present study defines an intermediate function, following the relationship-oriented approach, as the functional interdependencies between the function user (workers, equipment, or in-progress structures) and the provider (building products, temporary facilities, or site components).

Furthermore, several modeling methodologies have been proposed to represent various engineering behaviors or functionalities. There have been mainly two approaches: natural language (verb-noun pairs) and mathematical representation (transformation between input and output with respect to mathematical equations) (Chakrabarti and Bligh 2001; Deng 2002). Moreover, several studies have indicated that the functionalities of a product system are determined by the engineering behaviors of its constituent components, which depend greatly on their physical states (Keuneke, 1991; Ullman, 1993; Umeda et al. 1996; Qian and Gero, 1996).

Some studies indicate that function analysis can help incorporate manufacturing process requirements into the redesigns of manufactured products (Hayes, 1995). This implies that function analysis as well as function modeling can be used for improving constructability of a facility project with respect to evaluating intermediate function requirements. Several function analysis tools, like Value Engineering (VE) or Value Analysis (VA) (Sato, 2005; Fisk, 2003; Mukhopadhyaya, 2003; Younker, 2003; Dhillon, 2002) and Quality Function Deployment (QFD) (Lee

and Arditi, 2006; Eldin and Hikle, 2003; Ahmed et. al., 2003; Pheng and Yeap, 2001; Mallon and Mulligan, 1993), have been widely accepted by the manufacturing and construction industries. However, most of the existing function analysis tools are not suitable for abstracting and evaluating such construction requirements. The major reason is that they mainly focus on modeling the final usage functionalities with respect to the manufactured/constructed product. As such, they are inadequate to represent and evaluate the temporal attributes of intermediate function requirements associated with an in-progress facility product, whose physical states are frequently transited by construction processes.

## 2.3 Modeling Facility Product

Function modeling and analysis is closely related to product modeling. Specifically, the key process of QFD analysis compares the functionalities expected by the customers/clients with the product features that can be derived from a product model. The critical procedure of VA is to maximize the ratio of the expected values (functions) to the expenditure, which can be computed from the physical and cost attributes of the associated product components.

Several studies on facility product modeling have been conducted in the context of Computer-Integrated Construction (CIC) (Bjork and Penttila, 1989; Bjork, and Penttila, 1991; Bjork, 1992; Bjork, 1994). Most of these works focus on the methodologies to decompose a facility product and to structure the product modeling data. Meanwhile, several classification schemata have also been proposed in terms of terminology indexes for standardizing product decomposition. Some prevailing classification schemata are Uniclass (Crawford et; al., 1997), CI/SfB index (Alan and Wilfred, 1971), CAWS (Common Agreement of Work Sections for building works)

(Allott, 1998), NSB (National Building Specification), and Building (RIBA Enterprises, 2006). These classification schemata are typically based on similar hierarchical approaches defined by the associated terminology dictionaries. These classification schemata facilitate modeling a facility product by providing a common product decomposition hierarchy that may be easily referenced by fellow trades.

Such modeling languages as Object-Oriented language (Froese, 1996) and Express-G language (ISO/DIS 10303-11, 1992) have been applied to model facility products, and various representation schemata using these two languages have been proposed. Among these schemata, STEP (ISO/DIS 10303-1, 1993; Palmer, 1992) and IFC (IAI, 2004) are the most prevalent. However, most of these existing representation schemata as well as the product terminology indexes focus on modeling the completed facility project. Specifically, the physical attributes and the geometric attributes often describe the engineering characteristics of a product component in its completed state. Thus, the changing engineering characteristics of an in-progress product component along its construction life cycle are inadequately abstracted, leading to difficulties in using these existing product models to derive the temporal attributes of their engineering behaviors, which are necessary for evaluating the availability of an intermediate functionality.

Another important issue relating to facility product modeling is the integration capability of a product model to associate with other project perspective models (Bjork, 1992). A number of studies have proposed similar integration frameworks to link such project perspectives as product, process, resource, and contracting organization. Froese (1996) has presented a comprehensive review on many of these integration frameworks.

The integration of the product and process models forms the foundation for automatic schedule generation. This can be seen from a comprehensive comparison of various automatic schedule generation systems by Chevallier and Russell (1998). However, many of these automatically generated construction schedules may not have practical application. Some reasons, like the decomposition level of processes and construction method representation, have been put forward (Fischer and Aalami, 1996, Stumpf et al. 1996). Another possible cause, as a result of the present research, may be that the availability of an intermediate functionality that determines the commencement of the associated construction process is inadequately accounted for in previous works.

Previous integration frameworks use the construction state concept as the bridge between product and process models. Coupled with the similar concept of element activity, the component state concept enables the mapping of the construction processes onto the associated product components to depict the in-progress status of a product component (Zozaya-Gorostiza, 1989; Waugh, 1990; Jagbeck, 1994; Fischer and Froese, 1996; Luiten and Tolman, 1997). However, the construction state concept does not distinguish between the active state (when the associated construction activity is in progress) and the quiescent state (when no construction activity is performed on the in-progress product component). This leads to the deficiency in depicting the transitive engineering behavior of an in-progress product component. For example, the support-load functionality provided by an scaffold is only available when its erection has been completed.

Compared with the construction state concept, the component state concept distinguishes the engineering behaviors between the active and quiescent state phases (Chua and Song, 2002). This distinction is necessary to better represent the

intermediate function requirements. Component state describes both the changing material compositions and the engineering behaviors of a product component along its construction life cycle (Chua and Song, 2001; 2003). Additionally, the application of component state condition has been explored to integrate segmented schedules developed by different trades (Chua and Song, 2001; 2003). Furthermore, the component state concept can also be used to describe the intermediate function requirements and to integrate project modeling perspectives (Song and Chua, 2003).

## 2.4 Representation of Construction Sequencing Requirements

Many functional construction requirements for supporting construction processes can be described from the construction sequence viewpoint. A construction sequence indicates that the downstream activities may require functionalities realized in upstream construction. A large part of these functionalities are intermediate functionalities. The representation of construction sequencing requirements has been studied along two main research trends.

The first trend is that the construction sequencing knowledge can be represented as heuristic rules with various knowledge constructs and syntaxes (Navinchandra et. al., 1988; Morad and Beliveau, 1994; Dzeng and Tommelein, 1996). Most of these representation methodologies capture the sequencing rules from the process viewpoint as well as from the product viewpoint. They attempt to represent the reasoning logic in terms of cause-consequence relationships that associate the precedence relationships between processes and the functional interdependencies between product components. Such knowledge can be used for automatic schedule generation by deriving precedence relationships from product models.

Essentially, for example, a finish-to-start precedent relationship between an activity AX (constructing product component CX) and an activity AY (constructing product component CY) can be reasoned out from the functional interdependency that component CX supports component CY. Spatial factors such as topological relationships between or among product components (Chernef et. al., 1991) and installation directions (Morad and Beliveau, 1994) have also been incorporated into the heuristic rules for improving their ability to derive precedence relationships. However, the construction states of a product component, during which the component provides or requires some intermediate functionalities, have not been integrally considered in these representation schemata. This poses some difficulty in accurately describing the temporal availability of some functional interdependencies between in-progress product components.

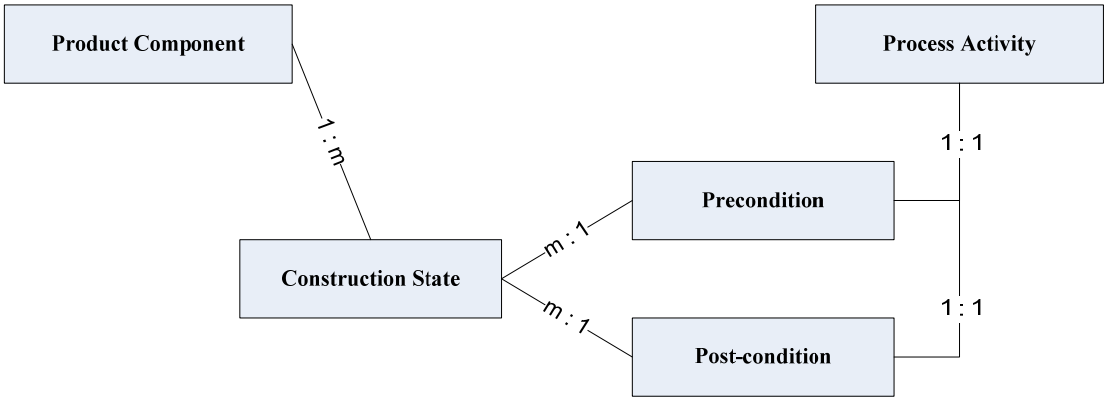



**Figure 2.1 Precondition and Post-condition of Construction Activity**

The second trend is that the construction sequence knowledge can be represented as the pre- and post-conditions of a class of activities in terms of construction states of product components (Froese, 1996; Luiten and Tolman, 1997). Figure 2.1 shows that the precondition, defining the trigger of the start event of the associated process, comprises a set of prerequisite construction states of the associated product components, while the post-condition defines the transitions to new

construction states of the product components worked on by the associated process upon its completion. This Figure also illustrates that one product component can have a number of construction states to describe the sequence of construction processes that work on the component.

However, the construction state (basic knowledge construct) does not further distinguish the active and quiescent state phases (Song and Chua, 2006). Accordingly, the differences of behavioral characteristics of a product component between its active and quiescent state phases cannot be captured by construction state conditions, so that the pre- and post-conditions may not be sufficiently defined in certain circumstances. Additionally, construction processes may not be the only cause for transiting the engineering behaviors of product component. The behavioral differences of a product component during the long duration of a natural chemical-physical process may not be adequately specified in the pre- and post-condition representation schema.

Most of previously developed construction sequencing knowledge can be used to derive only precedent relationships between construction activities. However, the non-precedent temporal relationships, especially concurrent relationships between construction processes or construction states, are difficult to be inferred using these existing knowledge representation schemata. The non-precedence relationships are crucial for evaluating intermediate function requirements.

## 2.5 Incorporation of Concurrency Relationships into Project Schedules

A major benefit of construction requirement analysis is to reduce the period of a construction schedule by better realization of the intermediate function requirements. Songer et al. (2000) have comprehensively reviewed several tools developed for

shortening a construction schedule. The major philosophy underlying these schedule reduction tools is concurrent engineering. Prevalent concurrent engineering approaches in the Architecture/Engineering/Construction (AEC) industry include Total Quality Management (TQM), participant partnering, and constructability review (de la Garza et. al., 1994), as well as design-built contracting and fast track development. Previous studies indicate that overlapping the sequential activities is a major strategy to advance project delivery (Chang and Ibbs, 1998; Eldin, 1997; and Prasad, 1996), especially for reducing design schedules (Bogus et. al., 2005).

Most of the previous studies have explored the overlapping relationships between two activities from the perspective of information dependency. The overlapping extent between two activities is plausibly affected by the degree of information dependency between them. Accordingly, four dependent types, namely independent, semi-independent, dependent, and interdependent, have been suggested to classify the information dependencies between the activities (Prasad, 1996). Other research focusing on reducing iteration loops in design schedule also implies that information requirement and availability should be a crucial factor that determines the overlapping relationships between the design tasks and construction activities (Chen et. al. 2003).

On the other hand, the risks or uncertainties arising from overlapping activities are also addressed with respect to shortening design schedules. A plausible risk is the unexpected iterative cycles and their negative impacts on downstream design activities where reworks may arise from frequent changes and unidentified errors in the upper-stream activities (Lee et. al, 2005). Some key uncertainty factors determining the overlapping extent have been studied. Specifically, information evolution and information sensitivity have been identified as two crucial factors that

determine the overlapping extent between two design activities (Bogus et. al., 2005). Other activity attributes such as task production rate, upstream task reliability, and downstream task sensitivity to upstream errors are also identified for planning and controlling a fast track schedule (Peña-Mora and Li, 2001).

These studies can help planners reduce project schedules by incorporating the overlapping strategies, and assist project managers in recognizing and proactively controlling the risks arising from overlapping tasks. However, from a practical viewpoint, the application of many existing overlapping strategies is limited to planning design schedules or programming as-built information required by designers, since the identification of concurrency relationships frequently focuses on the information requirement and availability. Thus, if the overlapping activity strategies can be extended to incorporate more types of construction requirements, the robustness of a project schedule would be further improved.

The intermediate function studies (Song and Chua, 2006; Song and Chua 2004) imply that the overlapping between/among component states as well as construction activities can be represented and evaluated with respect to intermediate functions. Particularly, the matching between the requirement and availability of an intermediate functionality imply a concurrency relationship. Meanwhile, the availability time-window of an intermediate functionality is also determined by the concurrency relationships among a set of component states. Consequently, the conflicts arising from improper concurrency relationships can be identified , leading to more robust concurrent schedules.

## 2.6 Modeling Space Requirements for Construction Processes

Space is an indispensable resource for constructing a facility product. Therefore, evaluating the fulfillment of space requirements in a construction schedule has been an important concern for many researchers. The space requirements have been studied in four main trends: site layout, workspace planning, 4D simulation, and time-space analysis.

Site layout studies focused on dynamic assignment of site spaces to temporary facilities and construction processes (Tommelein et. al 1992; Tommelein and Zouein 1993, 2001; Zouein and Tommelein 1999; Li and Love, 1998; Yeh, 1995; Hegazy and Elbeltagi, 1999). In general, a site space model needs to be integrated with a construction schedule for evaluating the dynamic layout of a site. Such evaluation is used for detecting the temporal collision among the site objects and for optimizing the transportation distances between the site space entities. Some artificial intelligence technologies, like Genetic Algorithm (GA) (Zouein and Tommelein 1999) and Neural Network (NN) (Yeh, 1995), have been explored for optimizing the dynamic layouts of a site along the construction period.

Previous studies on workspace planning have stressed that workspace should be considered as crucial resource for construction processes (Thabet and Beliveau, 1994a; Mallasi and Dawood, 2002). Additionally, the construction sequences are also constrained by occupation sequences of workspaces (Thabet and Beliveau, 1994b). Furthermore, the workspace requirements can be decomposed into some generic space utilization patterns, and these space utilization patterns have been categorized to facilitate detecting interferences among trades (Riley and Sanvido, 1995; Riley and Sanvido, 1997). In these studies, the workspace requirements from construction

processes as well as the placement requirements from product/resource components are often represented in 3D format.

4D simulation technology has been developed for visualizing the construction sequence as well as the temporal occupation and release of workspaces. In general, 4D simulation is realized through associating the 3D CAD models with the corresponding construction schedules (Mallasi, 2005; Soetanto, 2005; Chau et. al., 2004; Heesom et. al., 2003; Messner et. al., 2002; Koo and Fischer, 2000; McKinney and Fischer, 1997; Vaugn, 1996). Additionally, some studies employed 4D simulation models for facilitating site layout (Ma et. al., 2005; Soltani et. al., 2002; Tawfik and Fernando, 2001; Dawood and Marasini, 2000), and other studies explored the feasibility of using 4D simulation for recognizing the execution patterns of construction activities (Mallasi and Dawood, 2001; Mallasi and Dawood, 2002).

Based on the 4D simulation, several time-space analysis methodologies have been developed for detecting the time-space conflicts with respect to the temporal collisions (intersection topological relationships) between workspaces. Akinci et al. (2002a; 2002b; 2002c; 2003) proposed a pair-wise analysis between the workspaces occupied by two activities for detecting temporal collisions in a construction schedule. Guo (2002) detected spatial conflicts by categorizing the patterns of the space conflicts, and came up with a strategy to resolve the conflicts. This time-space analysis information can help construction planners improve construtability by alleviating space congestions and reducing trade interferences.

The common collision detection mechanism in previous studies is to combine the collision detection algorithm with the discrete event simulation method. This means that the collision detection computation between two work spaces may need to be implemented more than once, since the dynamic space world can only be

configured as a series of frames as the discrete event simulation proceeds. It is known that nearly all collision detection computation algorithms are very computationally intensive and resource consuming, implying that the existing time-space detection approaches may pose high demand for computation.

Song and Chua (2005a; 2005b) proposed the temporal space entity and temporal topological relationship concepts to represent a dynamic 3D space system for detecting spatio-temporal conflicts in a construction schedule, and also presented an improved approach for detecting time-space conflicts. This approach only performs the collision detection algorithm once, and the temporal intersection relationships can be derived by the comparatively time-saving Boolean operation on the existence periods of temporal space entities, which depict the durations when the associated space entities is occupied.

Additionally, only detecting temporal collision is inadequate for analyzing the spatial perspective of an intermediate function requirement. Some Geographic Information System (GIS) studies have suggested 8 types of binary topological relationships between two 2D regions (Egenhofer and Franzosa, 1990), and these eight types can be adopted to define the topological relationships between two 3D space entities (Song et. al., 2003). Particularly, the temporal ‘meet’ relationships has been used for detecting inaccessibility problems (Song and Chua, 2005). This implies that if non-intersection topological relationships can be incorporated into intermediate functions, more unfulfilled space utilization requirements can be detected through intermediate function analysis.

## 2.7 Comparison of Key Ideas of Present Research with Previous Studies

This section attempts to briefly compare the key ideas of the present research with the main contributions of the related previous studies. The present study develops the In-Progress Product Core Model (IPPCM) that extends the traditional product decomposition approach by a component state network that depicts in-progress configuration of a facility product, which is necessary for representing and evaluating intermediate function requirements. Component state developed in IPPCM represents the changing engineering behaviors of an in-progress product component. Based on this concept, the construction life cycle of a product component and the functional relationships between in-progress product components as well as the concurrent processing of a group of product components can be semantically represented with respect to interval-to-interval state relationships. This can help construction engineers identify more temporal relationships other than the precedent ones in CPM schedules.

Compared with the traditional construction state concept, a component state has both an active phase and a quiescent phase. This distinction facilitates differentiating the engineering behaviors associated with active and quiescent phases, leading to more accurate description of intermediate functions. Moreover, besides temporal attributes, a component state also has spatial attribute that references a number of 3D space entities in a space model.

The present study develops an analysis framework for evaluating intermediate function requirements. This framework comprises the intermediate function concept, the semantic model for representing intermediate functions, the schema for representing the intermediate function requirement knowledge, the information

integration framework for providing analysis data, and the analysis methodologies for evaluating intermediate function requirements.

The semantic model of intermediate functions provides an approach to evaluate intermediate function requirements using IT technologies. This semantic model incorporates the temporal and spatial attributes of the associated component states, which make it possible to evaluate the temporal and spatial perspectives of an intermediate function. Additionally, the ambiguities caused by the natural language representation can be largely resolved. Furthermore, the schema for representing the intermediate function requirement knowledge can be used to capture the temporal logics, especially those concurrent relationships, residing in construction methods. This knowledge can aid construction engineers in developing IPPCM and conducting intermediate functions analysis.

The information integration framework developed in the present study not only integrates the product and process modeling perspectives, but it also integrates another three modeling perspectives: resource, space, and intermediate function. These five modeling perspectives are necessary for intermediate function requirement analysis. Four analysis methodologies have been developed for analyzing an intermediate function from the temporal and spatial perspectives. This implies that the concurrent relationships, which are inadequately evaluated in traditional CPM schedules, can be evaluated with respect to the temporal relationships between/among component states.

Moreover, the existence vector concept and the corresponding Boolean operations are developed for implementing the temporal and spatial interaction analysis, which make it possible to evaluate these interactions without the need to conduct simulation of construction schedules. In this way, intermediate function

requirements can be timely and systematically communicated and analyzed, leading to improved constructability of facility projects.

# CHAPTER 3 IN-PROGRESS PRODUCT CORE MODEL

This chapter introduces the structure of the In-Progress Product Core Model comprising an extended product model for representing the decomposition of a facility product and a component state network for depicting the construction life cycles of product components and the functional interdependencies between in-progress components in terms of interval-to-interval state relationships. Moreover, the temporal and spatial attributes of a component state are also defined. Lastly, the Product Oriented Scheduling Technique (POST) is developed for deriving the temporal attributes of component states from the construction process schedule.

## 3.1 Structure of In-Progress Product Core Model

The present research develops an In-Progress Product Core Model (IPPCM) to characterize both the decomposition and the in-progress configuration of a facility product. Figure 3.1 illustrates the structure of an IPPCM comprising two parts. The left part of the Figure is an extended product model that represents the system decomposition of a facility product, while the right part is a component state network that describes the construction life cycle of the product components as well as the temporal relationships between the component states. The proposed IPPCM extends the product decomposition hierarchy that mainly describes the organization of a completed facility product by incorporating the construction life cycles of product components as well as the functional interactions between the in-progress product components. Additionally, besides the permanent product components, the temporary and site components are also incorporated into the extended product hierarchy.

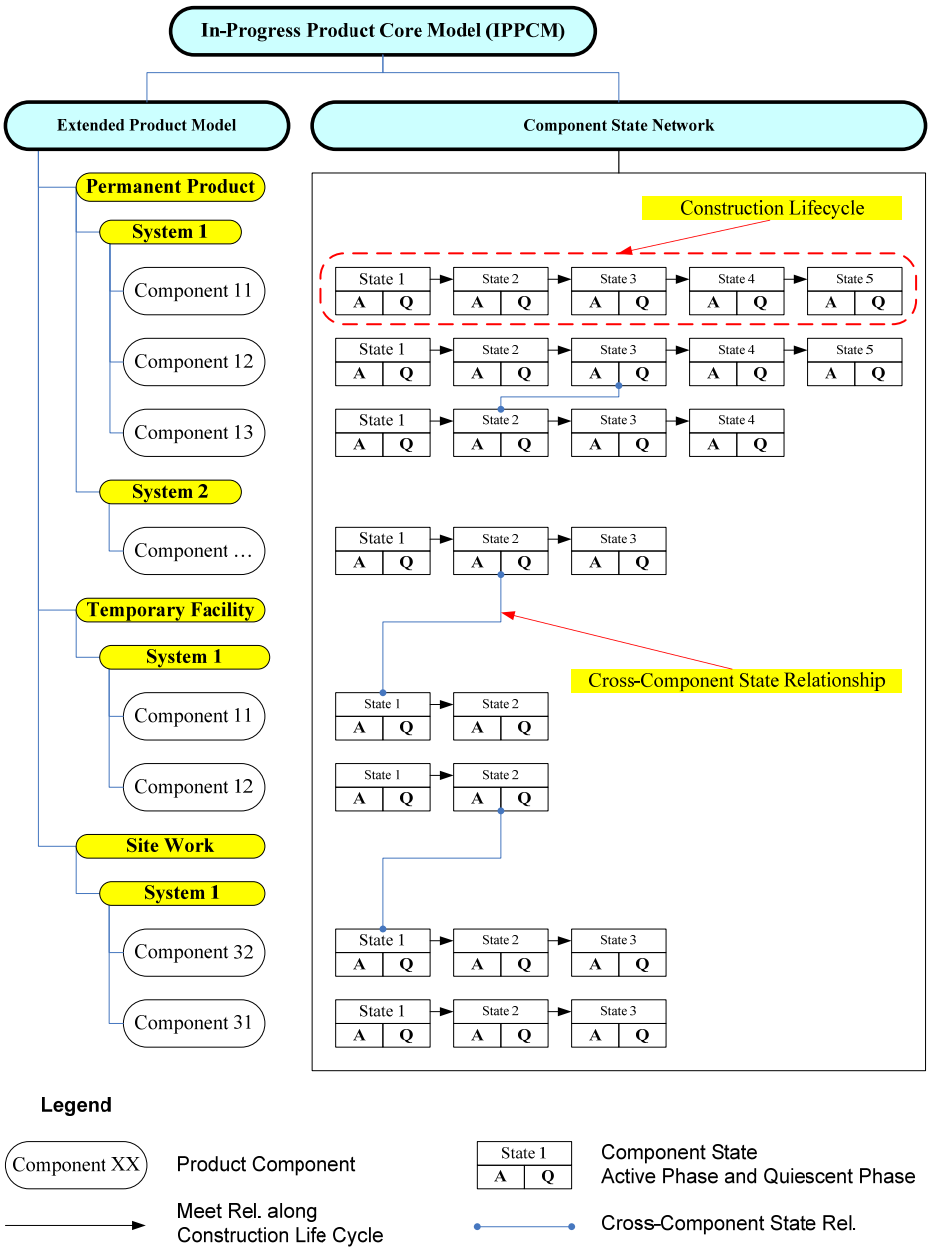



**Figure 3.1 Structure of In-Progress Product Model (IPPCM)**

## 3.2 Extended Product Model

### 3.2.1 Three Product Categories in Extended Product Model

Many existing product decomposition models mainly described the permanent facility product. The extended product model is essentially a product decomposition hierarchy that extends the traditional description of a permanent facility product by including both temporary facilities and site works for constructing the permanent

building. The permanent facility components of the extended product model depict the facility product that will be delivered to the project owner, while the temporary facility and site work components are not delivered to the project owner, but they are required to describe the construction requirements residing in various construction methods (Chua and Song, 2003). Therefore, the temporary and site components should be included in the product system to facilitate analysis of construction requirements.

The Extended Product Model (EPM) shown in Figure 3.1 illustrates these three product categories:

(1) ***Permanent Product***

A permanent product represents the permanent parts that will be delivered to the project owner after the construction. Once a permanent product component enters the product system, it will remain in the product system until the facility is demolished. A typical permanent building product normally comprises such subsystems as foundation, structure, enclosure, and HVAC systems.

(2) ***Temporary Product***

The temporary facility category comprises the temporary facilities that will be disassembled after the associated construction processes that require these temporary facilities are completed. Typical examples are formwork, brace work, scaffold, staging platform, and site accommodation facilities. In general, a temporary component will leave the product system after it is demolished or disassembled.

(3) ***Site Work***

The site work category comprises the site components that describe the site environment of the permanent facility, but do not belong to the permanent facility. Such site works as earth works and temporary accesses support construction processes by providing construction spaces, accesses, and suitable work environment.

### 3.2.2 Product Component

Figure 3.1 shows that each of the three categories can be further decomposed into systems, subsystems, and product components. The product component is the lowest level of product details that is necessary to describe the intermediate functions. In this regard, most of the facility components in the engineering design, like beams, columns, windows, and doors, can be modeled as product components in an extended product model. Sometimes, a large-size building part, like a long shear wall or a large-volume earth excavation, should be segmented into several parts. Such segmentation may be necessary to better describe the work package and facilitate analysis of construction requirements. In this way, each segment is represented as a product component. From a pragmatic viewpoint, the decomposition levels of different facility systems/subsystems can be determined by a construction engineer to a granulation degree that is suitable for evaluating the construction requirements.

## 3.3 Component State Network

A component state network comprises the sequences of component states that depict the construction life cycles of product components and the cross-component

state relationships that abstract the functional interactions or concurrent transitions between two in-progress product components.

### 3.3.1 Component State Concept for Depicting Construction Life Cycle

The present study develops the component state concept that describes the transitive engineering behaviour of a product component along its construction life cycle. This means that the entire construction life cycle of a product component can be divided into a sequence of component states. In this way, the construction life cycle of a product component can be abstracted from the engineering behavior perspective. Moreover, the engineering behaviors of a product component are often determined by its physical attributes, like material compositions, locations, and shapes, which are progressively altered by the construction processes associated with it. This means such a sequence of component states can also depict the sequence of construction processes executed on a specific product component.

Besides the construction processes, the engineering behaviors of a component may also be altered by some logistic activities processes. Specifically, the logistic activities may change the locations/shapes of product components. These changed geometric attributes of a component are significant for function analysis since they determine whether the component can be accessed by other component. For example, the functional space boundary of a portable platform is determined by its height and location.

A natural physical-chemical process, like hydration and drying which occur during curing, may not change the material composition or geometric attributes of a product component, but it may affect such engineering behaviors as acquired strength of a product component. Sometimes, the change in engineering behaviors along a

long-duration natural process need to be further distinguished by segmenting the duration of the natural process into several states. For example, the natural hydration process of concrete components can be segmented as three component states according to the “Setting”, “Hardening” and “28-Day Strength” stages.

### 3.3.2 Temporal Attributes of Component State

State Transition Points (STPs) are event time points that segment the construction life cycle of a product component into a sequence of component states, indicating that the STPs determine the temporal attributes of the associated component states. A major characteristic of these STPs is the markedly changed engineering behavior as well as material composition of a product component at these events. Accordingly, most of the start and finish events of construction processes can be treated as STPs. Moreover, the long-duration natural chemical-physical process, like hydration, can be segmented by a number of STPs to better reflect the changing engineering behaviors of a product component. In this way, the construction life cycle of a product component can be segmented by a sequence of STPs, and therefore its transitive engineering behaviors can be distinguished from state to state.

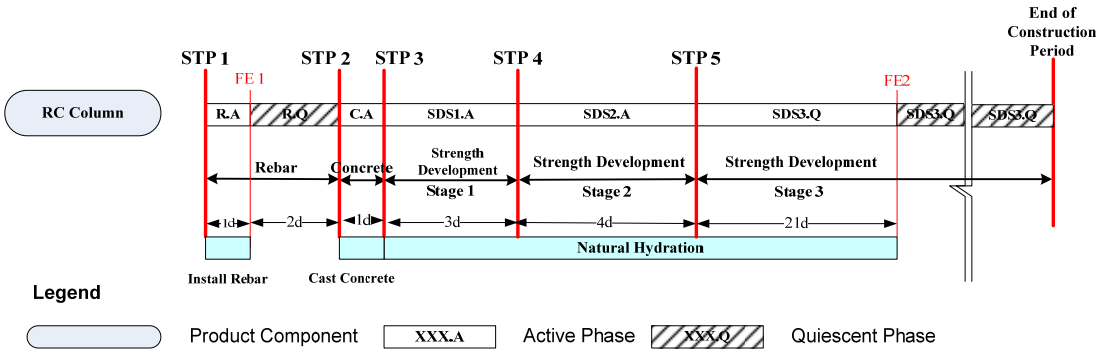



**Figure 3.2 Construction Life Cycle of RC Column**

Figure 3.2 shows that the “RC Column” component is realized by three sequential processes: “Install Rebar”, “Cast Concrete”, and “Natural Hydration”. Its

construction life cycle starts at “STP1”, which is the start event of the first construction process “Install rebar”, and finishes at end of the construction period. In general, the end of the construction life cycle of a permanent component is usually the end of the construction period since the permanent component will stay in the product system even after all the associated processes have been finished. In contrast, the end of the construction life cycle of a temporary component is often the finish time point of the associated “Disassembly” or “Dismantle” or “Remove” process since after this time point the temporary component leaves the product system.

Figure 3.2 also shows that the construction life cycle of the “RC Column” component is segmented into five sequential states by four STPs: “STP 2” to “STP 5”. “STP 2” is the start event of the second construction process “Cast Concrete”, while “STP 3” is the start event of the “Natural Hydration” process and also the finish event of “Cast Concrete”. The 28-day duration of “Natural Hydration” is further subdivided by “STP 4” and “STP5” into three states to depict the strength ranges of the in-progress “RC Column”, during which “RC Column” can provide the intermediate functionality for supporting different construction processes (see Figure 3.2). Specifically, after “STP 4” the column has acquired the 3-day strength and can maintain its shape by itself, while after “STP 5” it acquires its 7-day strength and can support the construction of components of the upper levels. Since “STP 5” the strength of the “RC Column” will continue developing for 21 days until the end of hydration process. In this way, the sequential changes of the engineering behavior of the “RC Column” component as well as its transitive material compositions are represented as a sequence of component states: Rebar → Concrete → Strength Development Stage 1 → Strength Development Stage 2 → Strength Development Stage 3.

Each component state is associated with two time points: *start* and *finish*, which determine the interval of the associated state duration. Figure 3.2, for example, shows that “STP 1”and “STP 2” are the *start* and *finish* time points of the state “Rebar”, respectively, and the corresponding state duration is 3 days. Additionally, if a component state is not the last state in the corresponding state chain its finish point is the same as the start point of its immediate successor state. For instance, “STP 2” is the finish point of the first state “Rebar” and also the start point of the second state “Concrete”.

Moreover, a component state can also be associated with an *active finish* time point that divides the duration into two phases: active phase and quiescent phase. Accordingly, the name label of a component state is marked by a postfix of either “.A” or “.Q” to differentiate its active and quiescent phases, respectively. Generally, an *active finish* time point can be the finish event of a construction/logistic/natural process. The active phase is the time interval between the *start* and *active finish* time points, during which the attributes of a product component are being actively altered by the associated process, whereas the quiescent phase is the interval between the *active finish* and *finish* time points, during which the component attributes are kept unchanged or can be modeled as unaltered from the engineering behavior perspective. In other words, an active phase is an active interval during which the associated process is in-progress, whereas a quiescent phase is a dormant gap between two consecutive activities performed on the associated product component, since the succeeding activity may not start just as the preceding activity finishes due to various reasons. If there is no gap between two consecutive processes executed on the same product component, the *active finish* time point is the same as the corresponding *finish* time point.

Distinguishing active phases from quiescent phases can facilitate the representation of intermediate functionalities. This distinction can better describe the engineering behaviors. Some engineering behaviors (intermediate functionalities) can only be realized when the associated product components are in their active phases, while other engineering behaviors can only be provided when their provider components are in their quiescent phases. As a result, this distinction can enhance the description of intermediate function requirements. For example, the temporary containing functionality can be provided when the formworks have been installed, which means that the formwork components are in their "Installed.Q" quiescent phase.

Figure 3.2 shows that the construction life cycle of "RC Column" has two *active finish* time points "FE 1" and "FE 2", which divided the "Rebar" and "Strength Development Stage 3" states, respectively. The corresponding two quiescent phases describes that the prevailing engineering behaviors during the two quiescent phases differ from their corresponding active phases. Firstly, "FE 1", which corresponds to the finish event of the activity "Install Rebar", is the *active finish* time point to segment the "Rebar" state into the active phase "Rebar.A" and the quiescent phase "Rebar.Q" (see Figure 3.2). During the 1-day "Rebar.A" state phase the "Install Rebar" process is being executed, while the "Rebar.Q" indicates the 2-day quiescent phase when the preceding "Install Rebar" has been completed but the succeeding "Cast Concrete" process has not started. The installation of the formwork can only be executed during "Rebar.Q" since it requires the required workspace that only exists when the "RC Column" is its "Rebar.Q". The second instance of quiescent state phase in Figure 3.2 occurs in the "Strength Acquirement Stage 3" state, indicating that the 28-day strength has been achieved at the finish event of the "Natural Hydration", i.e.

"FE 2" (Figure 3.2). Since then, the strength of the "RC Column" will remain unchanged until the end of construction period.

Figure 3.2 also shows that the "Concrete" state has a zero-duration quiescent phase since the "Natural Hydration" starts as soon as the concrete is cast. Meanwhile, both the "Strength Development Stage 1" and "Strength Development Stage 2" states have zero-duration quiescent phases since the "Strength Development Stage 1", "Strength Development Stage 2", and "Strength Development Stage 3" states are continuously transited by the "Natural Hydration" process.

It is important to note that the duration of a *quiescent phase* cannot be merely explained as the "free float time" in CPM schedules. The quiescent phase is determined by two activities sequentially executed on the same product component, whereas an activity may have several succeeding activities, and its free float time is determined by the succeeding activity that starts earliest. In this connection, the product component worked on by the activity may differ from that worked on by the succeeding activity that determine the float time.

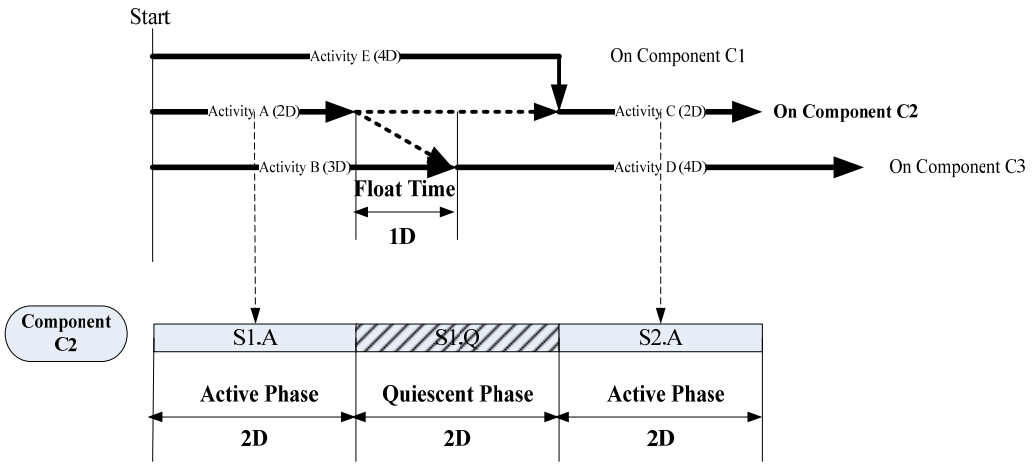



**Figure 3.3 Difference between Quiescent State Phase and Float Time**

For instance, Figure 3.3 shows that "Activity A" and "Activity C" are sequentially executed to transit the active phases "S1.A" and "S2.A" of "Component

C2", and there is 2-day quiescent phase "S1.Q" between "S1.A" and "S2.A". On the other hand, besides "Activity C", "Activity A" also has another successor "Activity D", which starts one day earlier than "Activity C". Consequently, there is a 1-day free float time for "Activity A", which is determined by "Activity D". Note that "Activity D" is executed on "Component 3", while "Activity A" is executed on "Component 2". This implies that some quiescent duration may not be derivable from a CPM schedule since it does not explicitly model the association relationships between activities and product components.

### 3.3.3 Spatial Attributes of Component State

The changes of the spatial characteristics of an in-progress product component can be represented by the state space attributes of each component state or state phase along it construction life cycle. In General, the *state space* attribute defines both the boundary of the physical shape and the location of an in-progress product component during the corresponding component state/phase. In this way, the geometric changes of a component along its construction life cycle can be depicted state by state.

Figure 3.4, for example, shows the shape changes of a site component "Access Road 2" along its construction life cycle. This site component will be excavated on Day 3, and kept open until Day 12 when it is backfilled. Accordingly, its construction life cycle can be segmented into a sequence of state phases: Original.Q → Excavated.A → Excavated.Q → Backfilled.A → Backfilled.Q, which is shown in the first column of the table in Figure 3.4. The top diagram in the Figure shows that the shape of the unexcavated "Access Road 2" is segmented into four space entities, namely "AR2_S1", "AR2_S2", "AR2_S3", and "TE3_S1", to facilitate the

description of its state spaces. The second column of the table illustrates the space entities referenced by the *state space* attribute of each state phase.

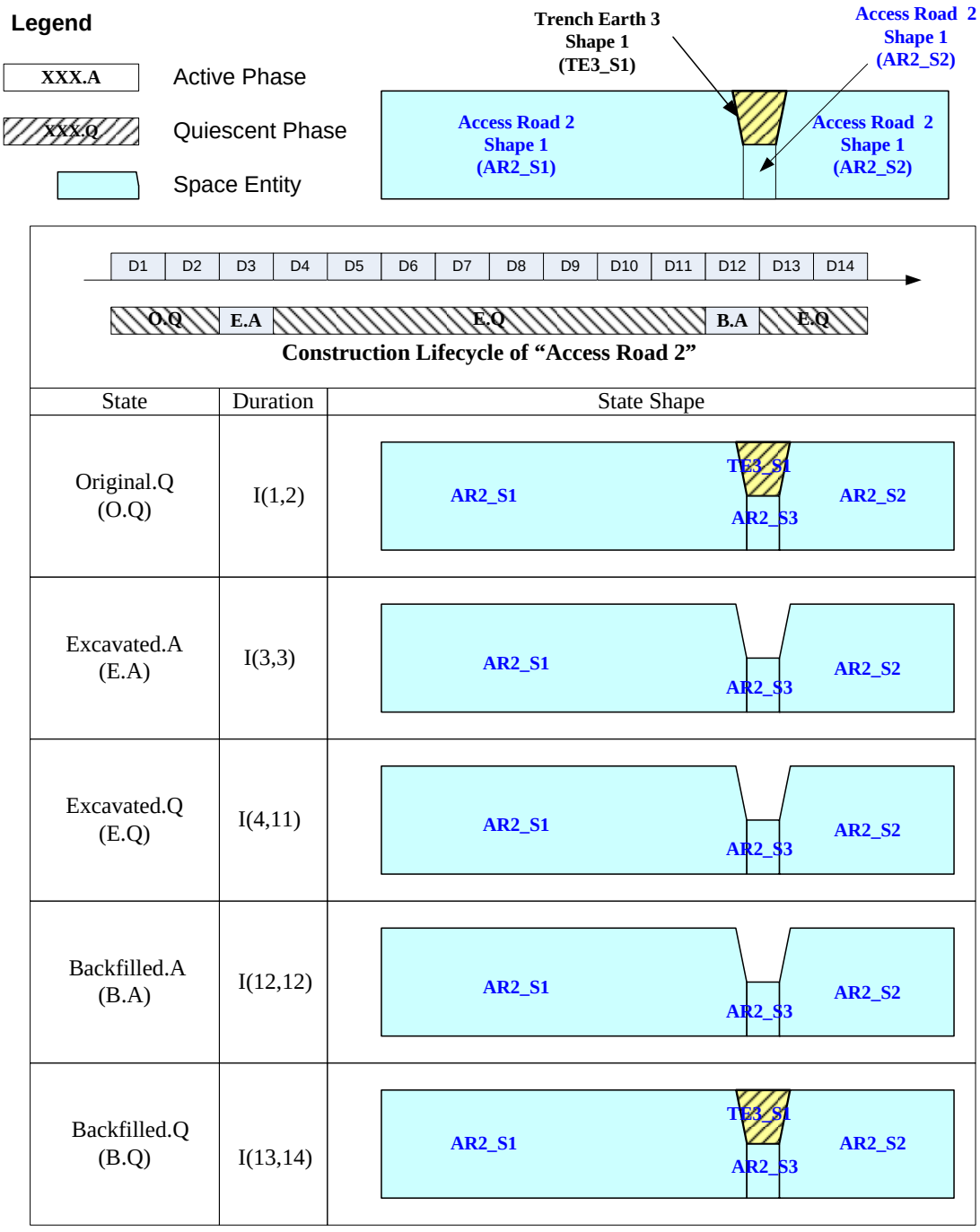



**Figure 3.4 State Space Attributes of State Phases of "Access Road 2"**

Specifically, the quiescent state phases "Original.Q" and "Backfilled.Q" reference the four space entities. On the other hand, the other three state spaces "Excavated.A", "Excavated.Q", and "Backfilled.A" do not reference the space entity

"TE3_S1" since the trench "TE3_S1" has been excavated and thus exits the space system during these state phases. The state space attribute of a product component is significant for evaluating the accessibility to its intermediate functionality. For example, when the trench in Figure 3.4 is kept open, a mobile crane cannot travel on it.

### 3.3.4 Interval-to-Interval State Relationships

The temporal relationship between two component states is the fundamental temporal logic used to depict both the construction life cycle of a single product component and the functional interactions between or among in-progress product components. Such temporal relationships can be represented in either a point-to-point format or an interval-to-interval format. The four types of precedence relationships used in the Critical Path Method (CPM) are represented in point-to-point format. In contrast, the present study employs the interval-to-interval format to depict the temporal relationships between the component states.

The temporal interval representation approach can provide greater flexibility and a richer semantic context to explicitly describe the precedent, coincident, coupling, and disjoint relationships between component states. The basis for this representation is the temporal interval algebra where the interval primitives and relationships have been formalized (Allen, 1984; Vilain et. al., 1990). Additionally, by using interval-to-interval format, the lag time in precedence relationships can be mapped onto one or more consecutive component states to give an explicit engineering reason for the lag time.

A time interval can be defined by two boundary points of time, namely the start and finish points, to represent a continuous interval. In the present study, "I($T_s$, $T_f$)" denotes a temporal interval that starts on $T_s$ and finishes on $T_f$. A temporal

interval relationship is a binary relationship that associates two intervals and is denoted by an interval relationship type. For instance, a temporal interval relationship "Before(iX, iY)", as shown in the first row of Figure 3.5, indicates that the finish point of the interval iX should be earlier than the start point of the interval iY. This further indicates that each point of time within iX is earlier than all the points of time in iY.

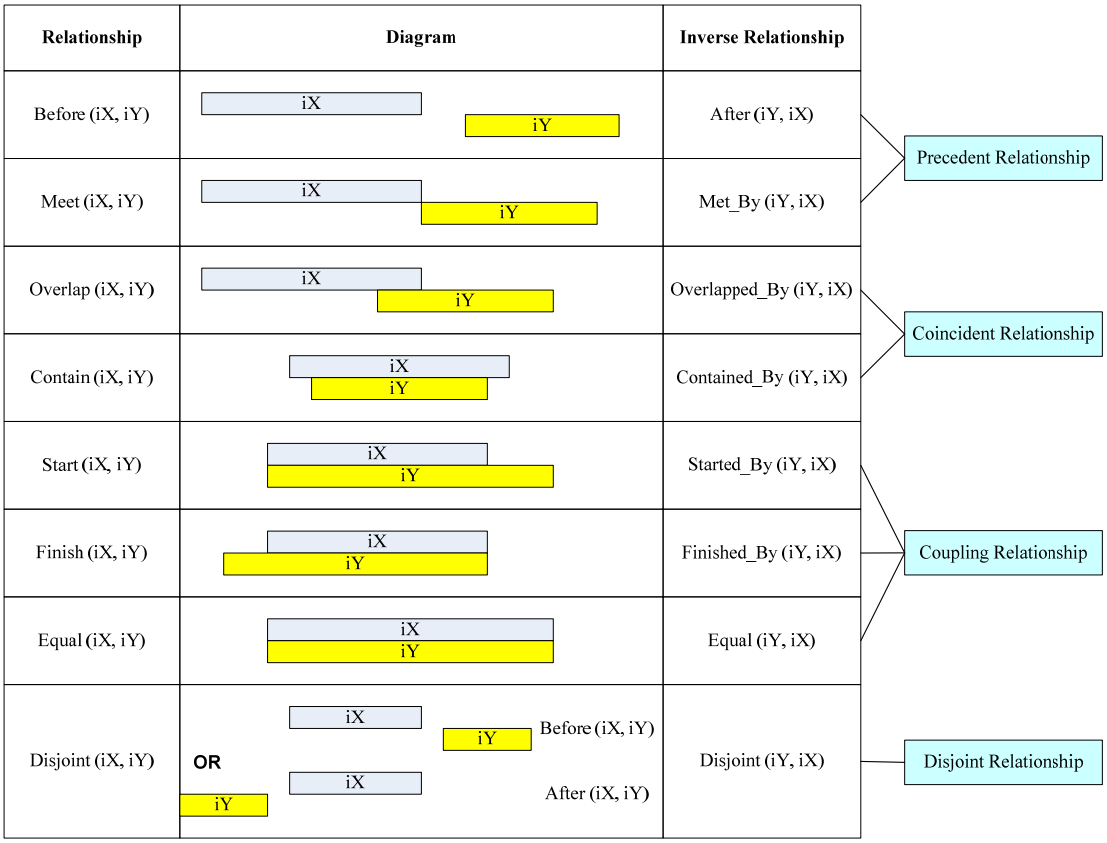



**Figure 3.5 Temporal interval relationships**

Figure 3.5 illustrates the thirteen types of temporal interval relationships proposed by Allen (1984), which have been widely referenced in many studies related to temporal logic representations and inferences. The inverse relationships between the corresponding types of interval relationships have also been included in the Figure. For example, "After" and "MetBy" are the inverse relationships of "Before" and "Meet" relationships, respectively. The present study further groups these thirteen

interval relationships into three categories, namely, precedent, coincident, and coupling as depicted in the Figure for better analysis of intermediate function requirements.

The interval-based precedent relationships comprising “Before”, “After”, “Meet”, and “MetBy” can be used to represent the sequence of the duration intervals of two component states, between which there is no overlapped interval. Moreover, a “Meet” relationship implies a “Before” relationship between two sequential intervals sharing one boundary point of time. The “Meet” relationship can be used to depict the sequence of states along the construction life cycle of a product component. “Meet” or “MetBy” relationships do not allow any temporal gap between two associated intervals, unlike Finish-to-Start precedence relationships in Activity-On-Node Networks that allow a lag time between the finish and start events of two successive processes.

Coincident relationships comprising “Overlap”, “OverlappedBy”, “Contain”, and “ContainedBy” relationships depict that one interval extends over another interval wholly or partially, but requires no simultaneous start or finish between the two associated intervals. The “Overlap” relationship can be used to describe the temporal availability of an intermediate functionality which is to be provided by the joint performance of a set of product components during certain states. Meanwhile, the “Contain” relationship can be seen as a stricter form of the “Overlap” relationship with a further condition that the two boundary points of one interval should lie within the other interval. The former relationship can be used to verify whether the temporal availability time-window (iX) of an intermediate functionality fully covers its requirement time-window (iY).

A stricter form of coincident relationship is depicted by the coupling relationship which comprises “Start”, “StartedBy”, “Finish”, “FinishedBy”, and “Equal” as shown in Figure 3.5. A coupling relationship defines the “Concurrency” relationships between the boundary points of the two state intervals, either the start point, finish point or both points. The coupling relationship can be used to describe situations when a set of processes are required to be executed simultaneously.

In the present study, another interval relationship “Disjoint” is specifically defined (see Figure 3.5). It is an important relationship for describing prohibited coexistence of two component states. For example, the painting state of a wall cannot be concurrently conducted with welding state of a pipe in an adjacent area. Mathematically, a “Disjoint(iX, iY)” relationship, indicating that there is no overlap between two intervals iX and iY, can be equally represented by the union of two precedent relationships, namely Before(iX, iY) and After(iX, iY).

Additionally, with temporal interval algebra, more temporal constraints can be inferred from a set of known interval relationships. For example, if it is declared that an interval iX “Contains” an interval iA, and iY “Contains” another interval iB, and the interval iA is “Equal” to the interval iB, the inference resultant from these three interval relationships is that iX should “Overlap” iY, indicating that these two intervals should overlap each other. This inference may help identify more temporal relationships for existing interval relationships, which can be used for analyzing intermediate functions as well as construction schedules.

### 3.3.5 State Relationships in Component State Network

The state relationships in a component state network can be abstracted from mainly four perspectives. Firstly, the construction life cycle of each product

component can be described by the “Meet” relationship to associate a preceding state with its immediate successor in a construction life cycle since the finish point of the preceding state and the start point of the immediate succeeding state share the same time point (state transition point). Secondly, the functional interdependencies between two in-progress components can be abstracted as the cross-component state relationships in terms of “Contain” or “ContainedBy” interval relationships. Thirdly, the concurrent construction of a group of product component in the same work package can be modeled by the coupling interval relationships like “Equal” or “Start” illustrated in Figure 3.5. Fourthly, some safety requirements can also be represented by disjoint interval relationships between the associated component states.

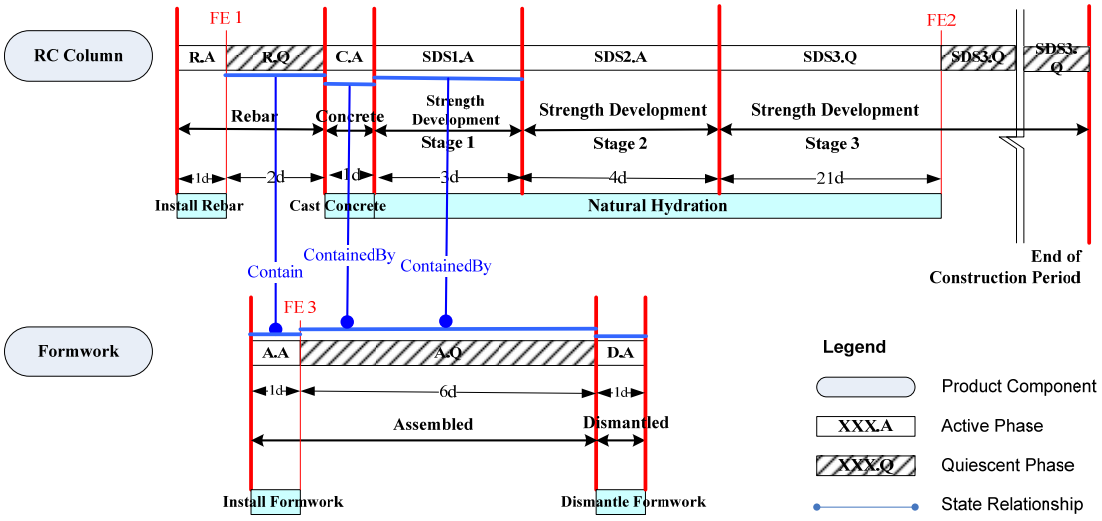



**Figure 3.6 State Relationships between Column and Formwork**

Figure 3.6, for example, shows the functional independencies between the in-progress “RC Column” component and the corresponding “Formwork” component. The construction life cycle of “RC Column” has been introduced in the preceding section, while the construction life cycle of “Formwork” is segmented into two sequential component states: “Assembled” and “Dismantled”. The former state is further decomposed into two state phases by the active finish “FE 3” corresponding to the finish event of the construction activity “Install Formwork”. The “Dismantled”

state does not have a quiescent phase since the temporary “Formwork” component leaves the product system after the “Disassemble Formwork” activity is finished.

Figure 3.6, for instance, shows that the “Rebar” state should “Meet” the “Concrete” state for the “RC Column” component along the construction life cycle of “RC Column”, while the “Assembled” state should “Meet” the “Disassembled” state along the construction life cycle of “Formwork”. In this way, the sequential transitions of product component states along its construction life cycle can be explicitly described with respect to “Meet” relationships between the immediately consecutive states.

Figure 3.6 also illustrates one “Contain” relationships and two “ContainedBy” relationships. The “Contain” relationships describes that the in-progress “RC column” provides the intermediate functionality for the in-progress “Formwork”, while the two “ContainedBy” relationships indicates that the in-progress “RC column” requires the intermediate functionality provided by the in-progress “Formwork”. Specifically, the quiescent phase “Rebar.Q” of “RC Column” should “Contain” the active phase “Assembled.A” of “Formwork”. This means that when the rebar of “RC Column” has been installed (“Rebar.Q”) and its concrete has not been cast, the in-progress column component provides work faces for executing the “Install Formwork” activity (“Assembled .A”). Figure 3.6 shows that “Install Formwork” lasts one day and can be conducted any time within the 2-day quiescent phase “Rebar.Q”. This also implies that the duration of “Rebar.Q” cannot be shorter than that of “Assembled.A”.

On the other hand, the “Concrete” and “Strength Development State 1” states of “RC Column” should be “ContainedBy” the quiescent state “Assembled.Q” of the “Formwork” component. This indicates that the concrete casting (“Concrete.A”) of “RC Column” requires the temporary support from the assembled “Formwork”

("Assembled.Q") and that the "Formwork" cannot be dismantled during the initial 3 days of hydration ("Strength_ Development_Stage_1.Q") in order to maintain the temporary stability of "RC Column". In this way, the functional interdependencies between two in-progress components can be depicted.

While reinforced concrete components and scaffold components have been exemplified for demonstrating that component state concept can be used for representing their construction life cycles and the functional interdependencies between the in-progress concrete and scaffold components. More examples like tendon, sliding formwork, traveling platform, access roads, trench works, scaffold, and falsework components will be presented later in Chapters 5 and 7 with respect to their state chains and cross-component state relationships. This implies that, in general, a sequence of states can be used for describing the construction life cycle and that cross-component state relationships can be employed for representing the functional dependencies between two in-progress components.

## 3.4 Product-Oriented Scheduling Technique (POST)

The Product-Oriented Schedule Technique (POST) (Song and Chua, 2003) is developed to derive the temporal attributes (start, active finish, and finish) of the states in the component state network. Using this scheduling approach, construction engineers can compute the duration intervals (state duration, active duration and quiescent duration) of component states as well as their state phases. These interval attributes are necessary for evaluating the fulfillment of intermediate function requirements. In this way, a construction schedule can be represented and analyzed from the product instead of the process perspective. Additionally, such a product-

oriented schedule may be referenced with ease by trades who mainly work on the product system, like designers.

### 3.4.1 Key Elements of POST

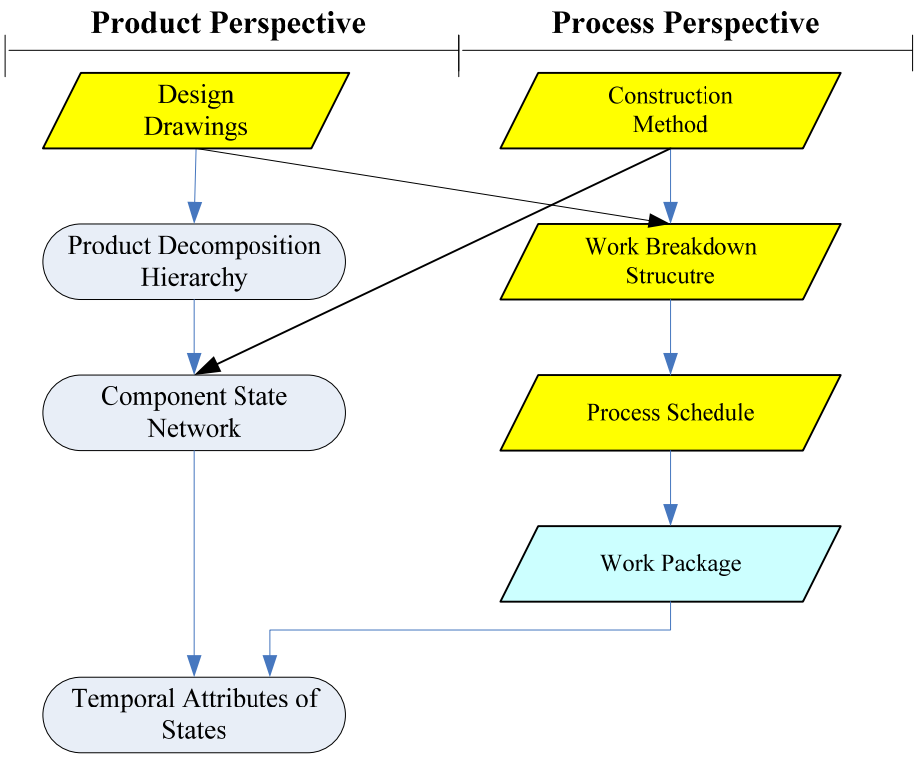



**Figure 3.7 Three Key Elements of POST**

Figure 3.7 shows the structure of POST comprising three key elements, namely product decomposition hierarchy, component state network, and temporal attributes of component states. As addressed in the preceding section, an extended product system can be first developed, and then a component state network can be developed to describe the construction life cycles of product components and the temporal relationships between the component states. Lastly, the developed component state network will be integrated with process schedule model via the work package concept, through which the temporal attributes of the component states can

be derived. In this way, the product model and the process model can be integrated from the scheduling viewpoint.

### 3.4.2 Work Package Concept

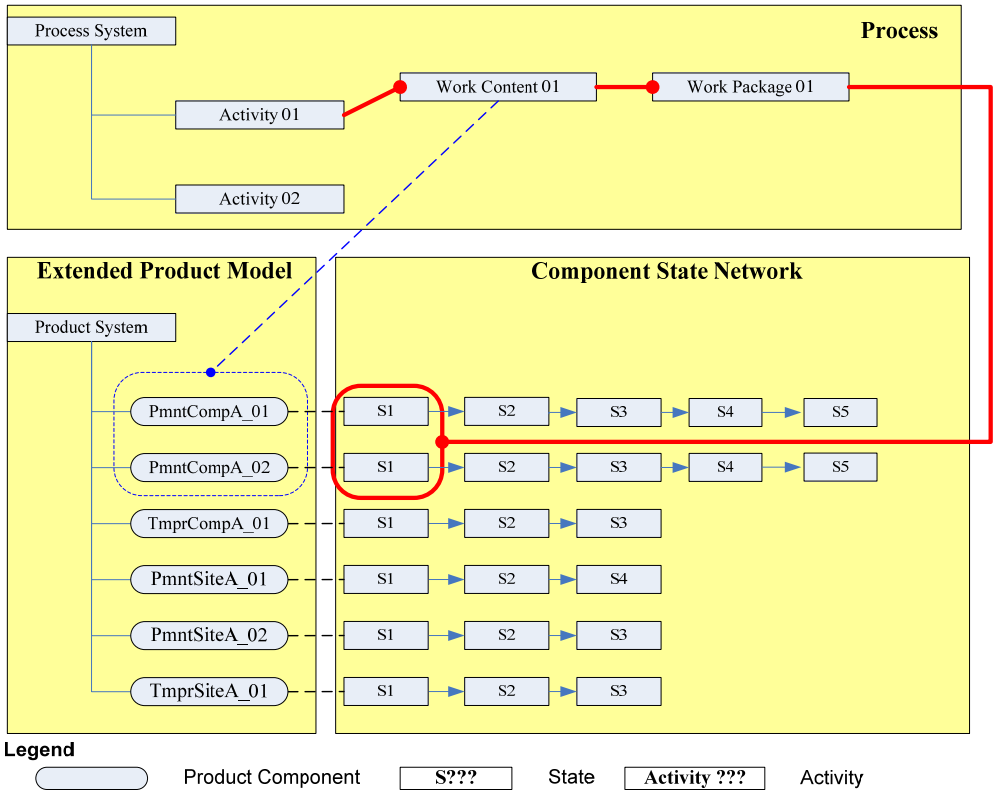



**Figure 3.8 Integrate Product and Process Models through Work Package**

Figure 3.8 shows the main reference relationships that associate process activities with component states through work packages. The Figure illustrates that a process can be associated with a number of product components via its work content attribute, which can be further described by the corresponding *work package* attribute. In general, a *work package* comprises a group of states (active phases) of the product components that are concurrently transited by the associated process. For example, "Work Content 01" of "Activity 01" comprises two permanent components "PmntCompA_01" and "PmntCompA_02", which will be constructed by "Activity 01", and "Work Package 01" comprises the "S1.A" active phases of

"PmntCompA_01" and "PmntCompA_02". In this way, through the work package a construction process in a process schedule associates with a number of component states in the corresponding component state network, which make it possible to derive the temporal attributes of component from the associated process schedule. Additionally, Figure 3.8 also implies that a component state network plays a pivotal role in integrating product and process perspectives.

### 3.4.3 Derivation of Temporal Attributes of Component States

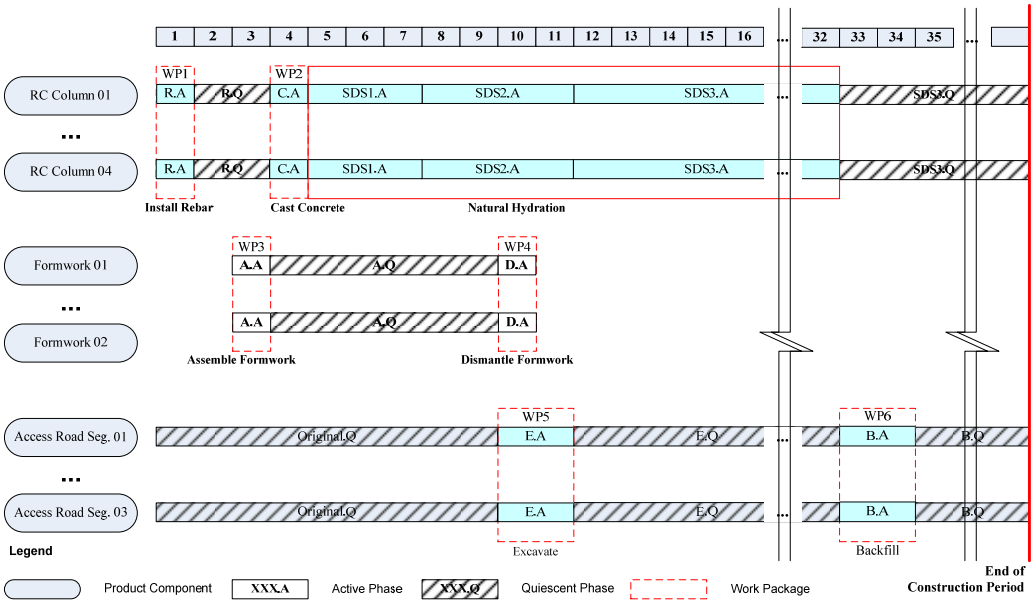



**Figure 3.9 Work Package for Deriving Temporal Attributes of Component States**

The derivation of the temporal attributes of component states depends on the nature of their transitions. In the case of a construction/logistic transition, the *start* and *active finish* time points of the active phases comprised in the work package are equal to the start and finish time points of the associated process. Figure 3.9, for example, shows a work package "WP1" comprising the active state phases "Rebar.A" ("R.A") of four components "RC Columns 01 to 04" associated with the activity "Install Rebar". Accordingly, the *start* and *active finish* time points of these four "Rebar.A" are on the start and end of Day 1, respectively, which are also the *start* and *finish* of

the associated process “Install Rebar”. Meanwhile, the *finish* time point of each “Rebar” state is equal to the start time point (start of Day 5) of its immediate successor state “Concrete” (“C.A”) comprised in the work package “WP2” associated with “Cast Concrete”. It can be deduced that the duration of the “Rebar” state is 3 days with a 1-day active phase and a 2-day quiescent phase. This is the case for all work packages of construction/logistic processes.

A component state can also be transited by a natural chemical-physical process. A natural process often starts as soon as the preceding construction activity is finished so that the component state transited by the preceding activity has a zero-duration quiescent phase, and the active finish and finish points of the preceding state as well as the start of the state transited by natural process are coincident. This is illustrated in Figure 3.9 by the “Natural Hydration” process which transits from “Concrete” (“C.A”) to “Strength Development Stage 1” (SDS1.A) of “RC Columns 01 to 04”. The Figure also illustrates the start of “Natural Hydration” is coincident with the finish (end of Day 4) of the preceding activity “Cast Concrete”. Thus, the start points of the four associated “Strength Development Stage 1” are on the end of Day 4. Particularly, the four “Concrete” states of “RC Columns 01 to 04” have zero-duration quiescent phases.

The duration attributes of nature-transited states are determined by the time required for completing the natural process, which is often independent of the quantity of work content. If a natural process is not further subdivided into several transition stages, the duration of the associated active phase is essentially the duration of the natural process. If the long duration of a natural process is segmented into several transition stages, the durations of the associated active phases are equal to the corresponding transition stages.

For example, Figure 3.9 shows that the 28-day duration of “Natural Hydration” is segmented into three transition stages, which last for 3 days, 4 days, and 21days, respectively. In this case, the *active finish* and *finish* time points of the states “Strength Development Stage 1” of “RC Columns 01 to 04” are on the end of Day 7. Accordingly, the active durations of these four states are 3 days. The time point and interval attributes of the “Strength Development Stage 2” and “Strength Development Stage 3” states can be likewise derived. Particularly, the “Strength Development Stage 1” and “Strength Development Stage 2” states have zero-duration quiescent phases, since each of the succeeding states begin as soon as the strength has been achieved. “Strength Development Stage 3” states have non-zero quiescent phases which start on the end of Day 32 (finish time point of “Natural Hydration”) and finish on the end of the construction period (see Figure 3.9), indicating that the 28-day strength has been achieved from the end of Day 32 (or start of Day 33) onwards.

There are two further special scenarios in the derivation of the temporal attributes of component states. Firstly, unlike a permanent component which stays in the product system after the last associated activity finish, a temporary component will leave the product system after it is dismantled or removed. Accordingly, the last state (“Dismantled” or “Disassembled”) in the construction life cycle of a temporary component should have a zero-duration quiescent phase. For example, Figure 3.9 shows that the “Dismantled” states of the two formwork components “Formwork 01” and “Formwork 02” have zero-duration quiescent phases.

Secondly, such site components as existing facilities and earth components exist in the product system from the start of the project. This scenario is represented by the initial states “Original.Q” with zero-duration active phases. Figure 3.9 shows the example that three site components “Access Road Segments 01 to 03” have 9-day

“Original.Q” quiescent phases since the first associated activity “Excavate” starts on Day 10. This means that the finish points of the three “Original.Q” quiescent phases are coincident with the start points (start of Day 10) of the succeeding “Excavated” states, respectively.

## 3.5 Concluding Remarks

The chapter develops the component state concept to represents the changing engineering behaviors of an in-progress product component. Based on this concept, the In-Progress Product Core Model (IPPCM) is developed to extend the traditional product decomposition model. In this way, the IPPCM not only has the capacity to represent the hierarchy structure of a completed facility product, but it can also describe the in-progress configuration of a facility product with respect to a component state network.

In such a component state network, the construction life cycle of a product component can be depicted by a sequence of component states, while the functional interdependencies between in-progress components are represented in terms of interval-to-interval state relationships. In this way, more types of temporal relationships, especially the concurrent ones, can be captured by using interval-to-interval relationships instead of point-to-point relationships as used in traditional CPM schedules.

A component state is described by both temporal and spatial attributes, which make it feasible to evaluate intermediate functions from both temporal and spatial perspectives. Furthermore, compared with the traditional construction state concept, a component state has both an active phase and a quiescent phase. This distinction facilitates differentiating the engineering behaviors associated with the two phases, respectively, leading to more accurate description of intermediate functions.

Additionally, Product Oriented Scheduling Technique (POST) is also developed for deriving temporal attributes of component states from process oriented schedules like CPM schedules.

It is of interest to discuss the mapping relationships between component states in a component state network and activities in a construction schedule. In general, a component state can be transited by either a construction process or a logistic process or a natural process. Provided that a process schedule comprises a completed list of construction activities, each construction process in the lowest decomposition levels of the process hierarchy can associate a set of component states in the component state network through their work package attributes. For example, the “Install Rebar” activity in the process schedule are associated with the “Rebar” states of a group of reinforced concrete columns on which “Install Rebar” will be executed (see Figure 3.2).

Similarly, the performer states of resource components like labor and cranes can be associated with the corresponding logistic activities in a process schedule. Some logistic activities may not be explicitly represented but implied by the construction activities, to which the resource components have been allocated. For instance, the activity executed by a crane to transport precast components may not be explicitly represented in a construction schedule, and it is often implicitly indicated by the construction activity “Launch Precast Component”. In this regard, the performer state “PS(Transport Precast Component)” of the crane can be associated with the construction activity “Launch Precast Component”.

Some component states are transited by natural processes that are often described as lag times of precedent relationships instead of activities in a construction schedule. Figure 3.2, for instance, illustrates the 3-day “Strength Development Stage

1” state transited by the natural hydration process. This state 3-day period is represented by the lag time of the finish-to-start precedent relationship between the “Cast Concrete” and “Dismantle Formwork” activities in a process schedule.

Furthermore, it is also meaningful to discuss the early/late times of an activity in relation to the associated component states with respect to the lengths of availability/requirement time-windows. In the present research, it is often assumed that the early start and finish times of a scheduled process determine the corresponding start and active finish time points of the associated states. In this regard, an in-progress product core model can be scheduled forward. On the other hand, the in-progress product core model can also be scheduled backward by mapping the late start and finish times onto the corresponding start and active finish time points of the associated states.

# CHAPTER 4 REPRESENTATION OF INTERMEDIATE FUNTION REQUIREMENT AND KNOWLEDGE

This chapter presents the intermediate function concept to represent functional construction requirements. Accordingly, a semantic model for representing functional construction requirements is also developed. Subsequently, a representation schema is developed for depicting intermediate function requirement knowledge from three perspectives. Lastly, an integrated information framework that integrates five project perspectives is developed for intermediate function analysis.

## 4.1 Characteristics of Intermediate Function

Intermediate functions form an important category of temporary functions that are required for supporting the construction of a facility project. In general, the intermediate functions are required for providing necessary work conditions, for performing construction processes, and for maintaining the stability and safety of an in-progress facility. The present study identifies some fundamental intermediate functions for supporting construction loads, maintaining stability of in-progress structures, providing workface, providing protection for finished works, and providing safe work environments. Some complex intermediate functions may comprise two or more fundamental functionalities. For example, providing a workable condition for painting skylights is a complex intermediate function requirement that comprises at least three fundamental intermediate functionalities: the temporary support provided by scaffold, the protection provided by tapes for the paneled glass, and the safe work environments that prohibit welding works in adjacent areas.

Although the purposes of intermediate functionalities may be different, they have two common characteristics that are different from usage functionalities of a facility product. Firstly, compared with most usage functionalities that are designed to last for the life cycle of the facility, intermediate functionalities are only required during a shorter period for supporting construction. Secondly, most usage functionalities are always available whenever they are required, whereas an intermediate functionality may only be available during certain period or periods since the engineering behaviors of an in-progress facility change as the construction processes as well as natural physical-chemical processes change its structures as well as the material compositions of its components. These two characteristics should be considered in modeling intermediate functions.

## 4.2 Semantic Model of Intermediate Function

### 4.2.1 Three Perspectives for Modeling Intermediate Function

A semantic model provides the constructors and designers an explicit description of an intermediate function so that they can have a common vehicle to communicate their intermediate function requirements. In this way, the intermediate function requirement knowledge can be captured from the previous project cases, communicated across trades, and reused with the aid of information technology. Furthermore, the semantic model makes it possible to explicitly incorporate the temporal and spatial logics that are embedded in the intermediate function requirements into the construction schedules. These temporal and spatial logics can then be evaluated for detecting unfulfilled intermediate function requirements. Consequently, the constructability of a facility can be improved.

However, it is difficult to apply most of the existing function analysis tools for representing intermediate functions. A major reason is that the function representation models employed in these tools lack the attributes to describe the temporal and spatial characteristics of the functions, and this greatly affects the evaluation of an intermediate function since both its requirement and its availability are time-dependent, as stated in the preceding section. The present study proposes a semantic model to describe an intermediate function from three perspectives: function user, function provider, and interaction between user and provider.

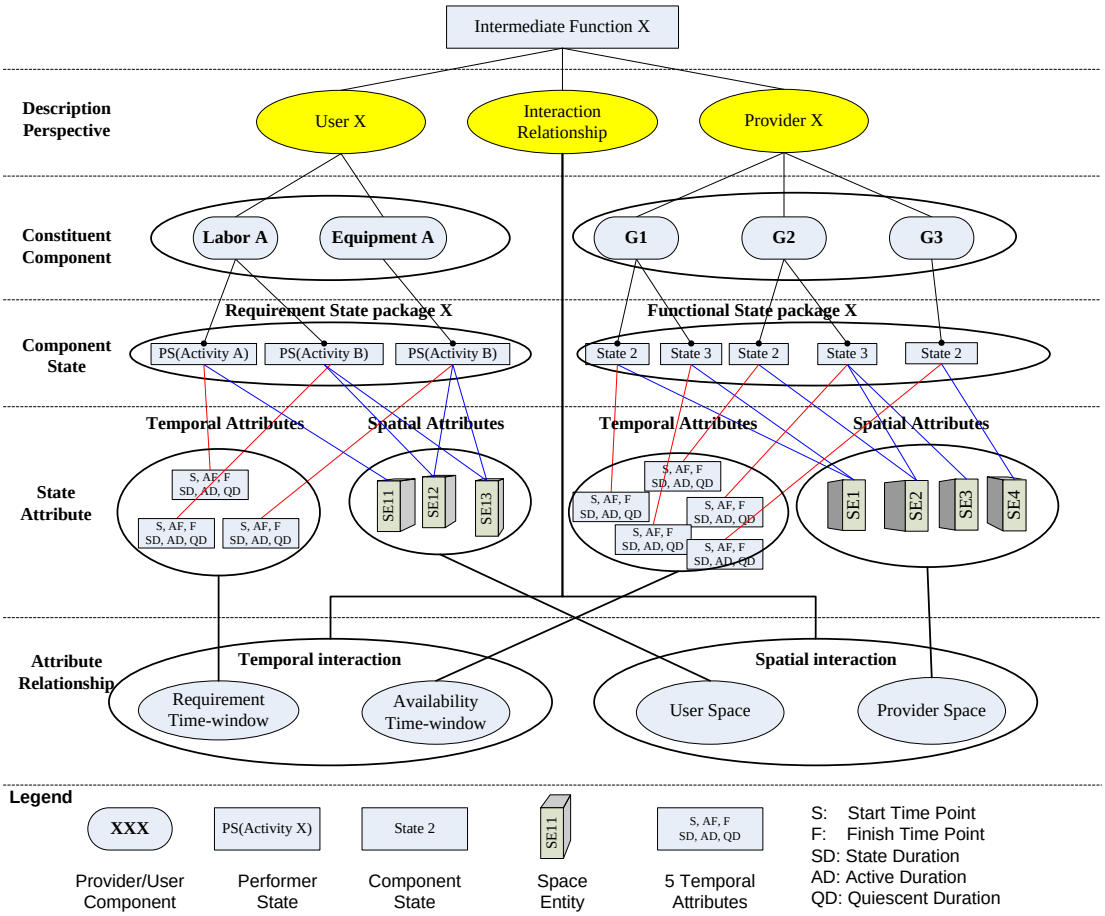



**Figure 4.1 Structure of Semantic Representation Model**

Figure 4.1 shows the semantic representation model comprising five layers. These five layers are description perspective, constituent component, component state, state attribute, and attribute relationship. Following this structure, a function user or a

function provider can be further decomposed into the component state level. The interaction between the user and the provider can then be assessed from both temporal and spatial perspectives since the component states associated with the user and the provider comprise both temporal and spatial attributes. In this way, the temporal and spatial logics embedded in an intermediate function can be semantically represented and then evaluated to ensure its fulfillment. The modeling elements in each layer will be explained in the following sections.

### 4.2.2 Function User and Requirement State Package

*Function user* is the requester of an intermediate functionality. It can comprise one or more in-progress product/resource components. Figure 4.1, for example, shows that the function user "User X" comprises two performer components "Labor A" and "Equipment A". A function user can be either a *resource-oriented* or a *product-oriented user*. A *resource-oriented user* comprises one or several construction performers (labors or construction equipment) requiring an intermediate functionality to support their loads, to access work faces, to acquire a work space, or to prevent themselves from hazards. On the other hand, a *product-oriented user* comprises a collection of in-progress components requiring an intermediate functionality to sustain their stability, to control their construction tolerances, or to protect themselves from damage.

As depicted in Figure 4.1, the construction state of a function user can be described by its *requirement state package*, which is a collection of states of the constituent user components, during which the user components would require the defined intermediate functionality. The associated states are called *requirement states*. For example, in Figure 4.1, the construction state of "User X" is depicted by

“Requirement State Package X” comprising three requirement states: the “PS(Activity A)” and “PS(Activity B)” states of “Labor A” and the “PS(Activity B)” state of “Equipment A”. This means that both “Labor A” and “Equipment A” would require “Intermediate Function X” when they are executing the corresponding “Activities A and B”. Particularly, the three requirement states only have active phases since the construction performers “Labor A” and “Equipment A” will leave the process system and not require “Intermediate Function X” after they complete the corresponding construction activities.

### 4.2.3 Function Provider and Functional State Package

*Function provider* represents the engineering solution for realizing an intermediate functionality. A function provider often comprises a group of in-progress product/resource components that jointly exhibit their behaviors to fulfill the defined intermediate function requirement. Similar to a function user, a function provider can be either a *resource-oriented provider* (labor and construction equipment) or a *product-oriented provider* (in-progress building system). For example, the “Provider X” in Figure 4.1 is a *product-oriented provider* comprising three product components “G1”, “G2” and “G3”.

Accordingly, the construction state of a function provider is depicted by its *functional state package*, which comprises the component/performer states during which the corresponding constituent provider components jointly behave to realize the defined intermediate functionality. The associated states are called *functional state*. Specifically in Figure 4.1, the functionality defined by the “Intermediate Function X” can be available only when the components “G1” and “G2” are in either “S2”state or

"S3" state and when the component "G3" is in its "S2" state. These five functional states are contained in "Functional State Package X".

Additionally, some components may simultaneously play both user and provider roles in two intermediate function systems. For example, a scaffold (provider) can temporarily support the workers (user) for installing skylight on the roof, and simultaneously it (user) should be supported by the underlying concrete structure (provider). The dual-roles played by some product/resource components imply that the in-progress components are interrelated through the functional interdependencies between them. This information may help planning engineers to trace the influence of the changes in designs and schedules with respect to the fulfillment of intermediate function requirements.

### 4.2.4 Temporal and Spatial Attributes of Component State

As addressed in the preceding chapter, there are three point attributes (Start(S), Active Finish (AF), and Finish (F)) and three interval attributes (State Duration (SD), Active Duration (AD), and Quiescent Duration (QD)) associated with each component state. The *state duration* interval is determined by the *start* and *finish* time points, and is divided into *active duration* and *quiescent duration* by the *active finish point*. The fourth layer in Figure 4.1 shows that each component state contained in the "Functional State Package X" is associated with these six temporal attributes. Meanwhile, each performer state contained in the "Requirement State Package X" has only active duration.

On the other hand, the *state space* attribute of each requirement/functional state can references one or several 3D space entity(ies). Figure 4.1, for example, illustrates that the state space of the requirement state "PS(Activity B)" of

“Equipment A” references two workspace entities “SE12” and “SE13”, while “SE13” is also referenced by another requirement state “PS(Activity B)” of “Labor A” to describe its workspace. Similarly, the provider component “G2” references “SE2” space entity in its functional state “S2” and references both “SE2” and “SE3” in the succeeding functional state “S3”, indicating that the physical space of “G2” is expanded as more material is added during the “S3” state.

### 4.2.5 Temporal and Spatial Interactions between User and Provider

Figure 4.1 shows that the functional interdependency between the function user and the function provider can be described by temporal and spatial interactions, in the fifth layer of the semantic model structure. The temporal interaction is described by the *requirement time-window* of the functionality required by the function user in relation to the *availability time-window* of the functionality provided by the function provider, while the spatial interaction can be evaluated with respect to the spatio-temporal relationship between the *user space* and the corresponding *provider space*.

The *requirement time-window* can be derived from the temporal attributes of the requirement states comprised in the requirement state package, while the *availability time-window* can be computed from the temporal attributes of the functional states comprised in the functional state package. The matching between the *requirement time-window* and the *availability time-window* can be used to verify whether the temporal interaction can be available in a construction schedule. The derivation of these two time-windows as well as the matching evaluation will be elaborated in Section 5.1.

An appropriate spatial interaction between the user and provider must exist in order for the user to utilize/access the intermediate functionality provided by the provider. The spatial interaction is depicted by the spatio-temporal relationship between the *user space* and the *provider space*. Figure 4.1 shows that the *user space* is a collection of space entities associated with the corresponding requirement states comprised in the "Requirement State package X", while the *provider space* comprises a set of space entities associated with the corresponding functional states comprised in the "Functional State package X". In the example of Figure 4.1, the *user space* comprises three space entities "SE11", "SE12", and "SE13", and the *provider space* comprises four space entities "SE1", "SE2", "SE3", and "SE4".

In this way, the spatio-temporal relationship between the *user space* and the *provider space* can be further described with respect to the twelve topological relationships that can be produced by pairing the three space entities in the *user space* and the four entities in provider space. These twelve topological relationships can be evaluated against the predefined spatio-temporal criteria for evaluating the availability of the spatial interaction. The evaluation methodology will be presented in Section 5.2.

## 4.3 Representation of Intermediate Function Requirement Knowledge

This section introduces a schema for representing the construction requirement knowledge from the perspective of the intermediate function. This knowledge representation schema comprises a set of basic knowledge constructs and a collection of relationships to associate these constructs. These knowledge constructs and association relationships can be used for scripting the domain knowledge according to the syntax rules specified in the schema. Specifically, this schema employs two basic

knowledge constructs defined from the product perspective, namely (product) component type and (component) state type. The association relationships use the temporal interval relationships stated in Chapter 3 for representing the functional interdependencies among component states. The syntax rules will be defined with respect to intermediate functionality from three perspectives, namely construction life cycle of single product component using *state chain type*, functional interdependencies between two in-progress product components using *state interaction type*, and availability condition of an intermediate functionality provided by a group of product components using *intermediate function availability type*. Consequently, such a schema can be used to capture the temporal logics residing in construction methods. These temporal logics can facilitate the evaluation of the availability of intermediate functionalities and also help establish the state relationships in a component state network.

### 4.3.1 Two Basic Knowledge Constructs

#### *4.3.1.1 Product Component Type*

Product component instances of the same type have similar functional and material characteristics, and are often constructed using the same construction method(s). The product components can be categorized into different categories, i.e. *component types*, according to such component attributes as function/role, material composition, geometric attributes, and physical attributes. These attributes will be explained in the following.

Firstly, *name label* represents the name of the product component type which is used as the type identifier for constructing the intermediate function requirement knowledge. Secondly, *function/role* indicates the functional role that the component

will play in the completed facility product system. In other words, it describes the purpose for which the component has been designed. For example, such structural components as columns, beams, and slabs are designed for supporting various loads. Thirdly, *material composition* indicates the main material constitution of a product component which often implies the possible construction methods to build the component. For example, a concrete component can be constructed using precast or cast-in-situ method. Fourthly, the *geometric attributes* attribute indicates shape, location, surface area, and volume of a product component. These geometric attributes determines the interaction between product components as well as the work content.

Lastly, *physical properties* include such physical attributes as rigidity, moisture, dryness, and elasticity. Besides the *material composition*, these *physical properties* may also determine the behavioral characteristics of a product component. Additionally, these properties are often changed by the natural chemical-physical processes like drying and hydration. This is the main rationale for mapping the effects of natural processes onto the corresponding state(s) of a product component.

#### *4.3.1.2 State Type*

Throughout the construction life cycle of a product component, its material composition and physical attributes as well as its geometric attributes may change, leading to corresponding changes in its engineering behaviors. These changes in behavioral characteristics can be abstracted by a sequence of component states divided by a series of state transition points. The component state instances of the same type often indicate that their associated in-progress components can perform similar engineering behaviors for realizing certain intermediate functionalities. The *state type* can be defined as a category token for characterizing both the compositional

and behavioral natures of a collection of in-progress components of the same component type. Accordingly, a group of component states of the same type would display similar behavioral characteristics, and would be associated with similar types of construction processes that transit the material and geometric attributes of the associated in-progress components.

The semantic representation of a state type can be defined by four attributes. Firstly, *name label* represents the name of the state type which is used as an identifier for constructing the intermediate function requirement knowledge. Secondly, *in-progress material composition* indicates the constituent material added to (or deducted from) the component in the construction process (denoted by active state phase). Thirdly, *related/associated process type* indicates the construction or logistic or natural chemical-physical process that contributes to the defined state. Fourthly, *in-progress physical characteristics* (like rigidity, moisture, dryness, and elasticity) may determine the transitive engineering behavior (like strength, and adhesion) of an in-progress product component. Thus, an in-progress component may exhibit different engineering behaviors or play distinct functional roles at different states along its construction life cycle.

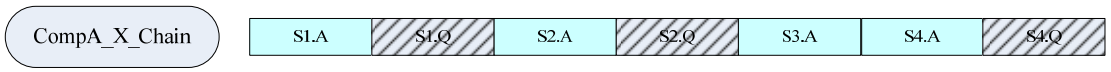
### 4.3.2 Three Representation Syntaxes

#### *4.3.2.1 State Chain Type*

If product components of the same type are built using the same construction method(s), their construction life cycles would follow the same sequence of state transitions, though the state durations may differ depending on the work content. Accordingly, the entire construction life cycle of a product component can be characterized by a state chain type describing the sequential changes of its

engineering behavior, which meanwhile reflects the sequence of the construction, logistic, or natural processes contributing to these changes.

The representation syntax of a typical *state chain* type is illustrated in Figure 4.2, where "CompA_X_Chain" represents the construction life cycle of component type "CompA" built using the construction method "X". The *state sequence* is the critical attribute defining the sequence of state types delimited by "{" and "}". The component state types, separated by semicolons, are listed in temporal order. The temporal interval relationship between each two consecutive states is the "meet" type (see Figure 3.5 for the complete set of interval relationships). For example, state "S1" "Meets" state "S2" (see Figure 4.2).




**Type Name:** CompA_X_Chain
**Component Type:** CompA
**Construction Method:** X
**State Sequence:** {S1; S2; S3.A; S4}

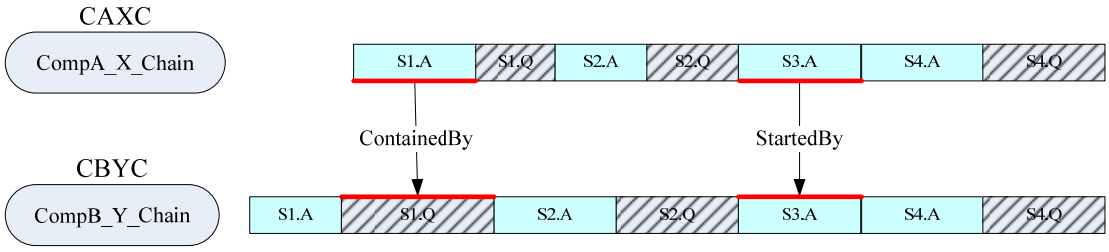
**Figure 4.2 Representation of state chain type**

Some states in the *state sequence* marked by a postfix ".A", like state "S3.A" in Figure 4.2, indicates that this state does not have a quiescent phase in the construction life cycle. This is possible in a natural chemical-physical process. For example, the hydration period of a concrete column can be separated into two states, "3-day strength" and "7-day strength" states. The end of the former, with no quiescent phase, indicates that the concrete strength has developed sufficiently to maintain its own shape, while the end of the latter active phase indicates that the strength has been further developed to provide the functionality to support the weight of the upper structure components. Such distinctions in the state chain may be necessary for

depicting the evolving intermediate functionalities that may be required by different downstream processes.

From the functional interdependency viewpoint, the preceding state of a product component often provides the intermediate functionality required by the succeeding state along its state chain. For example, the bottom and side segments of a box girder should be constructed before its top segment, since the top segment requires the support functionality provided by the bottom and side segments. Additionally, the state sequence of the mobile component, like a sliding formwork, a traveling platform, or a portable platform, can also be determined by the sequence of its changing locations or workspace provision.

#### *4.3.2.2 State Interaction Type*




**Type Name:** CompA_X_Chain and CompB_Y_Chain
**1st State Chain Type:** CompA_X_Chain (CAXC)
**2nd State Chain Type:** CompB_Y_Chain (CBYC)
**State Interaction:**
{ContainedBy(CAXC.S1.A, CBYC.S1.Q);
StartedBy(CAXC.S3.A, CBYC.S3.A)}

**Figure 4.3 Representation of state interaction type**

The *state interaction* type depicts both the functional interdependencies and concurrent transitions residing in the construction method(s) with respect to the temporal relationships between the state chains of two product components. Its representation syntax is illustrated in Figure 4.3, which demonstrates the cross-

component state relationships between two state chains “CompA_X_Chain” (*first state chain*) and “CompB_Y_Chain” (*second state chain*).

The critical attribute *state interaction* comprises a collection of temporal interval relationships established from one state in the *first state chain* to another state in the *second state chain*. Generally, each temporal interval relationship can be either a coincident or a coupling relationship. The coincident relationship, like “Contain” or “ContainedBy”, depicts the functional interdependency between the two in-progress components, while the coupling relationship types depict some concurrent transitions of their states (either concurrent start, or concurrent finish, or both).

The example in Figure 4.3 shows one functional interdependency and one concurrent transition in terms of temporal relationships between the states of “CompA_X_Chain” and “CompB_Y_Chain”. The functional interdependency is represented by a coincident relationship given by “ContainedBy(CAXC.S1.A, CBYC.S1.Q)” which indicates that the intermediate functionality required by the “CompA” component during its active phase “S1.A” is to be provided by the “CompB” component during its quiescent phase “S1.Q”, and this intermediate functionality does not exist when “CompB” begins its transition to the “S2.A” active phase. For instance, the interaction “ContainedBy(Beam.Concrete.A, Formwork.Assembled.Q)” would indicate that the concrete of the beam (function user) can be cast only when the corresponding formwork (function provider) has been assembled (quiescent state phase “Assembled.Q”) before it is dismantled (active state phase “Dismantled.A”).

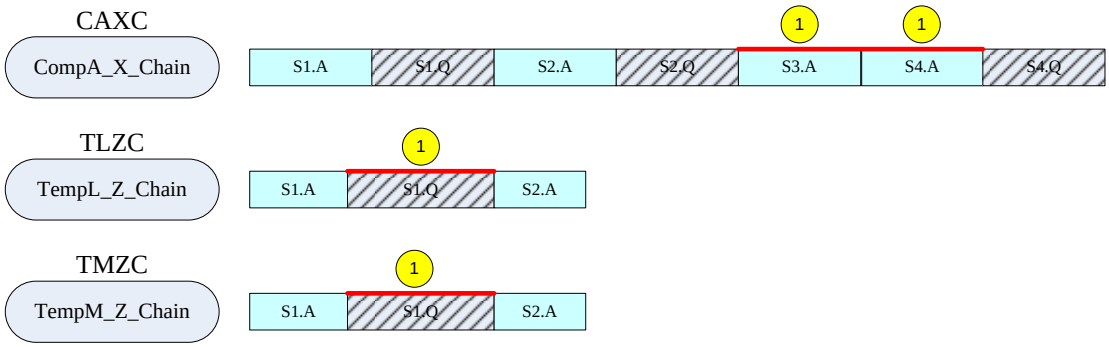
On the other hand, the concurrent transition relationship is represented by a coupling relationship given by “StartedBy(CAXC.S3.A, CBYC.S3.A)”, which denotes that the active transition of state “S3” of “CompB” simultaneously triggers

the active transition of the “S3” state of “CompA”. For instance, “StartedBy(Slab.Concrete.A, Beam.Concrete.A)” indicates that concrete slab should be cast simultaneously with the underlying beams for structural integrity.

#### *4.3.2.3 Intermediate Function Availability Type*

In general, an intermediate functionality is provided by the function provider comprising a group of components of various component types in certain component states/phases. These states/phases are termed as functional states/phases, which determine the availability of the intermediate functionality. During these functional states, the associated product components contribute their engineering behaviors to the realization of the defined intermediate functionality. Therefore, not only should the type of components to realize the intermediate functionality be depicted, but the component states/phases that determine the availability of the intermediate functionality should also be explicitly defined.

The intermediate function availability type is used to abstract the necessary condition to realize the intermediate functionality. The representation syntax of an intermediate function availability type is illustrated in Figure 4.4. The most important attribute is the availability condition that is delimited within the “{” and “}”. This condition references a set of functional states, and some provider components may have more than one functional state. The example in Figure 4.4 illustrates the availability condition representing the joint performance of the provider components in their functional states to realize the “iFunctionX” functionality. The functional states are marked by the circled one in the Figure.




**Type Name:** iFunctionX
**Functionality Description:** Temporary support functionality
**Referenced State Chains**: CompA_X_Chain (CAXC);
TempL_Z_Chain (TLZC); TempM_Z_Chain (TMZC).
**Availability Conditions:**
1 { CAXC[S3.A; S4.A]; TLZC[S1.Q]; TMZC[S1.Q]};

**Figure 4.4 Representation of intermediate functionality type**

The condition "{CAXC[S3.A; S4.A]; TLZC[S1.Q]; TMZC[S1.Q]}" requires that the component "CompA" be in its two active phases "S3.A" and "S4.A" and that both temporary components "TempL" and "TempM" be in their quiescent state phases "S1.Q" so that these three types of components in an in-progress structure jointly provide the "iFucntionX" functionality. Additionally, the "S2" state of either "TempL" or "TempM" does not have the quiescent phase, since the components are being disassembled during the active phase of the "S2" state and they leave the product system after the disassembly is completed.

## 4.4 Information Integration Framework

### 4.4.1 Structure of Information Integration Framework

In Chapter 3, the Product Oriented Scheduling Technique (POST) indicates the integration of the product and the process models through the work package concept. This integrated product and process model can be extended to cover more project perspectives.

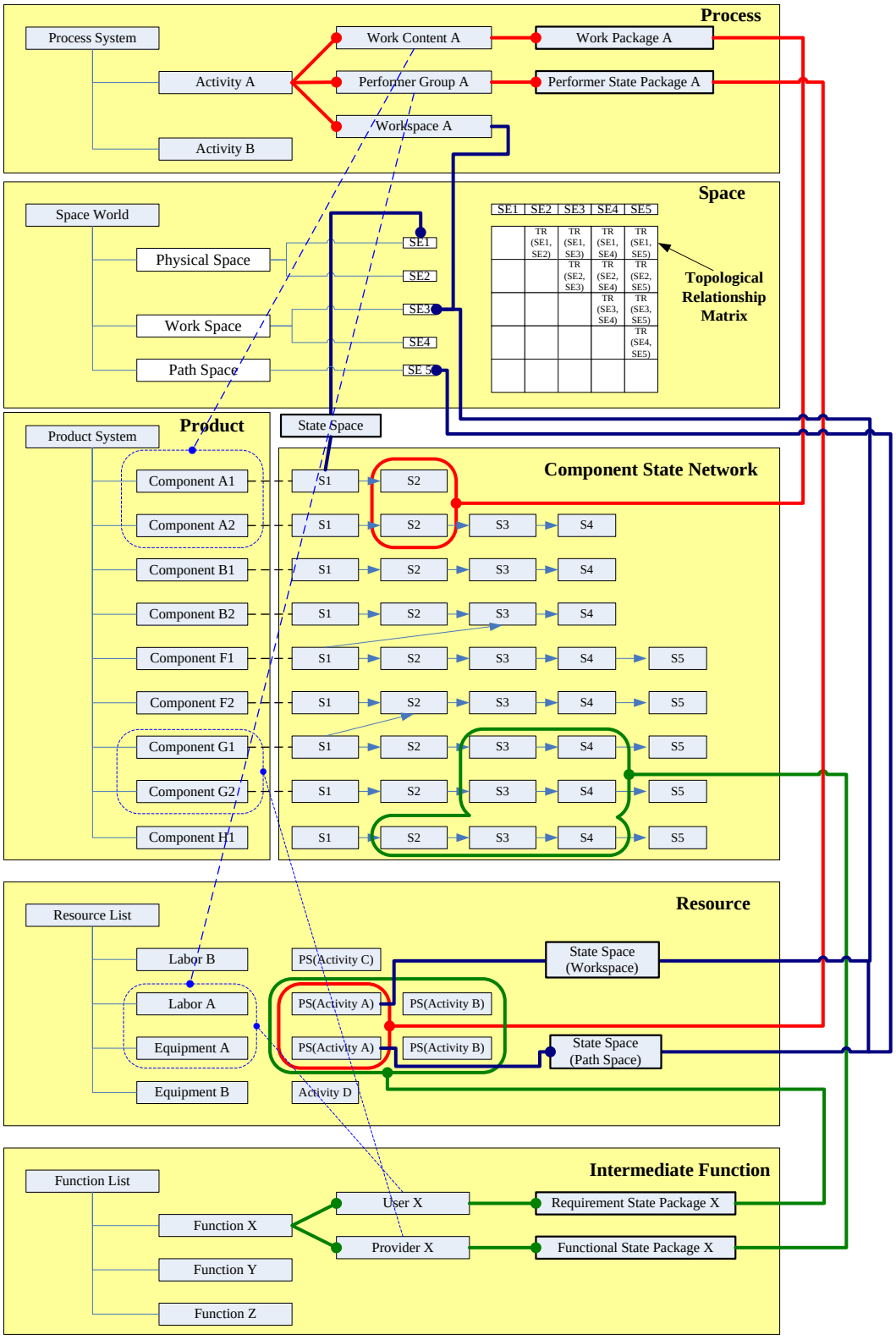



**Figure 4.5 Structure of Information Integration Framework**

Figure 4.5 depicts the information integration framework that associates the five project modeling perspectives, namely product, process, resource, space, and

intermediate function. These five modeling perspectives are required for analyzing intermediate function requirements. The integration of the five perspective models is achieved through four types of state package concepts and two space reference attributes. Besides the work package, the other three are performer state package, requirement state package, and functional state package as depicted in. Figure 4.5 also shows the space model referenced by the process and product as well as resource models through the workspace attribute of process and state space attribute of component state. Moreover, the Figure also shows that the component state network acts as the kernel to integrate four project perspectives: product, process, space, and intermediate function.

### 4.4.2 Space Model

The space model describes the geometric characteristics of a product system and also characterizes various space utilization requirements arising from executing construction processes, transporting equipment and materials, and storing materials. The space model comprises a hierarchy of space entities and a triangle matrix to describe the topological relationships between the space entities, as shown in Figure 4.5.

"Space World" represents the root of the space hierarchy system, which can be decomposed according to different utilization purpose into such space subsystems as physical space, work space and path space subsystems, which themselves can be further decomposed into embedded subsystems until space entities in the lowest hierarchy levels. Briefly, the space boundary of a space entity/subsystem in the low hierarchy level should be contained or enclosed by the associated higher-level ones.

The space entities in the lowest hierarchy levels can be represented by the 3D solid or surface objects in a 3D CAD model.

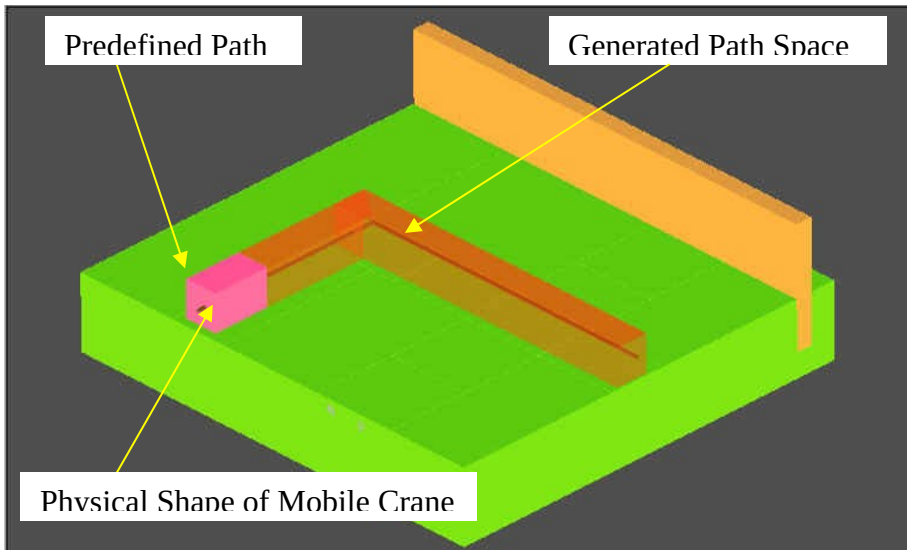



**Figure 4.6 Generation of Path Space of Mobile Crane**

A workspace subsystem or entity describes the boundaries within which the associated construction labors and equipment can execute their construction tasks. For example, the workspace subsystem of the reinforced concrete structure of an 8-floor building can be decomposed into 8 floor zones, and each floor zone can be further decomposed into 3 work zones, each of which can be further detailed into several workspace entities. Similarly, the trajectory boundary required by mobile equipment and portable facilities, like mobile cranes, can be represented by path space entities, which can be generated in a 3D CAD model by extruding the physical shape of a mobile equipment/facility along predefined tracks/paths (see Figure 4.6). Also the physical spaces occupied by the permanent building components can also be structured according to the product system hierarchy. In this way, most of the space utilization requirements during construction can be abstracted.

Furthermore, the spatial relationships between two space entities can be described in terms of (binary) topological relationships. In the present study the

topological relationships are simply categorized into three types, namely *disjoint*, *intersection*, and *meet*. The *disjoint* relationship means that there is no surface or volumetric overlap between two space entities, while the *intersection* relationship indicates that there is overlapped volume between space entities. The *meet* relationship defines that two space entities only share some common surfaces, but there is no overlapped volume between them. The *intersection* topological relationships can be used to detect temporal collisions among workspaces (Song and Chua, 2005), while the present study focuses on employing the *meet* relationships for evaluating the spatial interaction between function user and function provider.

Figure 4.5 shows that the topological relationships between a pair of space entities can be represented as a triangle matrix, since the reverse topological relationship from Space Entity Y to X is the same as the corresponding topological relationship from Space Entity X to Y in the context of the present categorization scheme of topological relationships. The triangle matrix in Figure 4.5, for example, only records the topological relationship "TR(SE1, SE2)" from space entities "SE1" to "SE2", since the reverse topological relationship TR(SE2, SE1) is the same as "TR(SE1, SE2)".

### 4.4.3 Work Package and Performer State Package

An activity in the process system module of Figure 4.5 can be described by two important attributes: the *work content* and the *performer group*. The former refers to the product components that will be processes by the activity, while the latter denotes the labor and equipment allocated for executing the activity. "Activity A" of Figure 4.5, for example, has "Work Content A" (referencing "Components A1 and A2" in the product model) and "Performer Group A" (referencing "Equipment A" or

"Labor A" in the resource model). These two attributes can be further described from the viewpoint of construction state by the work package and performer state package, respectively.

Figure 4.5 shows that "Work Content A" is depicted by "Work Package A" comprising the states "S2" of "Components A1 and A2". The active phases of these two "S2" states are transited by "Activity A". Meanwhile, the "Performer Group A" of "Activity A" can be further described by the "Performer State Package A" comprising the *performer states* "PS(Activity A)" of "Labor A" and "Equipment A". In this way, the temporal attributes of "Activity A" can be mapped onto the two performer states for further intermediate function analysis.

Additionally, if a construction performer is assign to execute several activities, its execution history can be described by a sequence of performer states corresponding to the activities. This is similar to the construction life cycle of product component, but the sequence of performer states may be discontinuous since there are gaps between the construction activities along the execution history of the associated construction performer. Figure 4.5, for example, shows that "Labor A" is firstly allocated to perform "Activity A" and then "Activity B" so its execution history comprises two sequential performer states: "PS(Activity A)" and "PS(Activity B)".

Consequently, the integration of the process and product models can be realized through the work package concept, while the integration of the process and the resource model can be established by the performer state package. Then, the temporal attributes of construction activities can be mapped onto the component/performer states for deriving their temporal attributes. These derived temporal attributes will be used for evaluating the temporal interaction of an intermediate function.

### 4.4.4 Requirement and Functional State Packages

As stated in the preceding Section 4.2, the *requirement state package* comprises a set of component/performer states during which the function user would require the intermediate functionality for supporting the construction processes or for maintaining the stability of an in-progress structure. On the other hand, the availability of the intermediate functionality is depicted by the *functional state package*, comprising a collection of component/performer states during which the constituent components jointly exhibit their engineering behaviors to realize the intermediate functionality.

Figure 4.5 illustrates the intermediate function “Function X” that is associated with resource model through “Requirement State Package X”. The “Requirement State Package A” comprises the performer states “PS(Activity A)” and “PS(Activity B)” of both “Labor A” and “Equipment A”, which would require the “Function X” functionality when they are executing “Activity A” and “Activity B”. Likewise, the intermediate function “Function X” is also associated with component state network through “Functional State Package X”, marked by a rounded rectangle in Figure 4.5. This means that the “Function X” functionality is available only when “Components G1 and G2” are in either state “S3” or “S4” and when “Component H1” is in the state “S2”, “S3” or “S4”. These seven component states are contained in “Functional State Package X”.

### 4.4.5 Workspace and State Space

The information integration framework shows that a process model can be integrated with a space model via its workspace attribute. Besides the work content

and performer group attributes, an activity has another attribute workspace, which can reference a number of space entities in a space model to describe the space boundary required for executing the activity. Figure 4.5, for example, shows that “Workspace A” of “Activity A” references the space entity “SE3” to describe the space boundary required for executing “Activity A”.

The state space of a component state may reference one or several space entities in a space model to describe the physical space occupied by the in-progress component. Figure 4.5, for example, shows that “S1” state of “Component A1” references the space entity “SE1”. Similarly, the state space attribute of a performer state can also reference several space entities to describe the workspace where the construction performer executes the associate activity. For example, the performer states “PS(Activity A)” of both “Labor A” and “Equipment A” reference the same space entity “SE3”, indicating they work together within the same workspace. This means that the workspace of an activity can be mapped onto the associated performer states as their state spaces.

Additionally, Figure 4.5 also shows that besides “SE3”, the performer state “PS(Activity A)” of “Equipment A” references another space entity “SE5”, which describes the path space for transporting “Equipment A” to the work face. Consequently, the spatial attributes of component/performer states can be used for evaluating the spatial interaction between the function user and the function provider, which is elaborated in Section 5.2.

## 4.5 Concluding Remarks

The semantic model of intermediate function provides a common vehicle to represent and communicate intermediate function requirements among the trades

distributed in different engineering domains, leading to reduced ambiguity of understanding intermediate function requirements across trades. Moreover, this semantic model also makes it feasible to manage intermediate function using information technologies.

A schema has been developed for representing the intermediate function requirement knowledge from three perspectives, namely the construction life cycle of a single product component, the functional interdependencies between two in-progress components, and the availability conditions for a group of provider components. This knowledge capture the temporal logics residing in construction method, which can aid planning engineers in developing IPPCM and modeling intermediate function requirements.

Lastly, an information integration framework is developed for integrating five project modeling perspectives: product, process, resource, space, and intermediate function, which are necessary for intermediate function analysis. The integration is realized through work package, performer state package, requirement state package, and functional state package as well workspace and state space. Thus, the information integration framework creates the foundation for developing various analysis methodologies in the succeeding chapter.

# CHAPTER 5 INTERMEDIATE FUNCTION ANALYSIS METHODOLOGIES

The semantic representation model of an intermediate function indicates that an intermediate function can be analyzed from both temporal and spatial perspectives. Based on this semantic representation model, four analysis methodologies have been developed. The first and second methodologies are used for detecting unavailable temporal and spatial interactions of a single intermediate function, respectively. The third methodology extends the first to match the requirement time-windows with the availability time-windows of a set of compatible/substitutable intermediate functionalities. The fourth methodology is developed to identify the bottleneck state that determines the earliest availability of an intermediate function, which aids planning engineers in advancing bottleneck states for early realization of the associated intermediate functionalities, leading to earlier commencement of construction activities and reduced construction periods.

## 5.1 Evaluation of Temporal Interaction between User and Provider

### 5.1.1 Computation of Requirement Time-Window

The temporal interaction between the function user and the provider can be evaluated by matching the *Requirement Time-Window* (RTW) of the user with the *Availability Time-Window* (ATW) of the provider. The *requirement time-window* of an intermediate function defines the time interval(s) during which the intermediate functionality will be required by the function user, and this time-window can be derived from the associated requirement states (or state phases). In general, a *requirement time-window* should cover the duration intervals of all the requirement

states comprised in the requirement state package RSP(F). Mathematically, such a time-window can be derived by the Boolean union operation of the duration intervals of all the associated requirement states or state phases, given by:

$$\mathbf{RTW(F)} = \bigcup_{\mathbf{i,j}} \left( \mathbf{I}\left( \mathbf{C_i.S_j} \right) \right) \quad \forall\ \mathbf{C_i.S_j} \in \mathbf{RSP(F)} \tag{5.1}$$

Where *F* is an arbitrary intermediate function, *RTW(F)* the requirement time window of *F*, $C_i.S_j$ the $S_j$ state of the $C_i$ component, $I(C_i.S_j)$ the duration interval of $C_i.S_j$, and *RSP(F)* the requirement state package of the intermediate function *F*.

For example, Figure 5.1A shows the function user "User_1" of "Function_1", comprising two construction performers "R1" and "R2" along with their requirements states "R1.Act_01", "R1.Act_02", "R2.Act_01" and "R2.Act_02" comprised in the requirement state package "RSP(Function_1)". Accordingly, the temporal attributes of two "Act_01" and two "Act_02" states can be derived from the associated construction activities "Act_01" and "Act_02" through the corresponding performer state packages "PSP(Act_01)" and "PSP(Act_02)", respectively (Figure 5.1A). In the Figure, the two "Act_01" states are equal to each other, and the two "Act_02" states are also equal to each other, indicating that "R1" and "R2" are jointly executing the construction activities "Act_01" and "Act_02".

Accordingly, the requirement time window RTW(Function_1) can be derived from the duration intervals of the four performer states, which is given by the Boolean union operation as follows:

**RTW(Function_1)**

**= I(R1. Act_01) ∪ I(R1.Act_02) ∪ I(R2.Act_01) ∪ I(R2. Act_02)**

**= RI1 ∪ RI2**

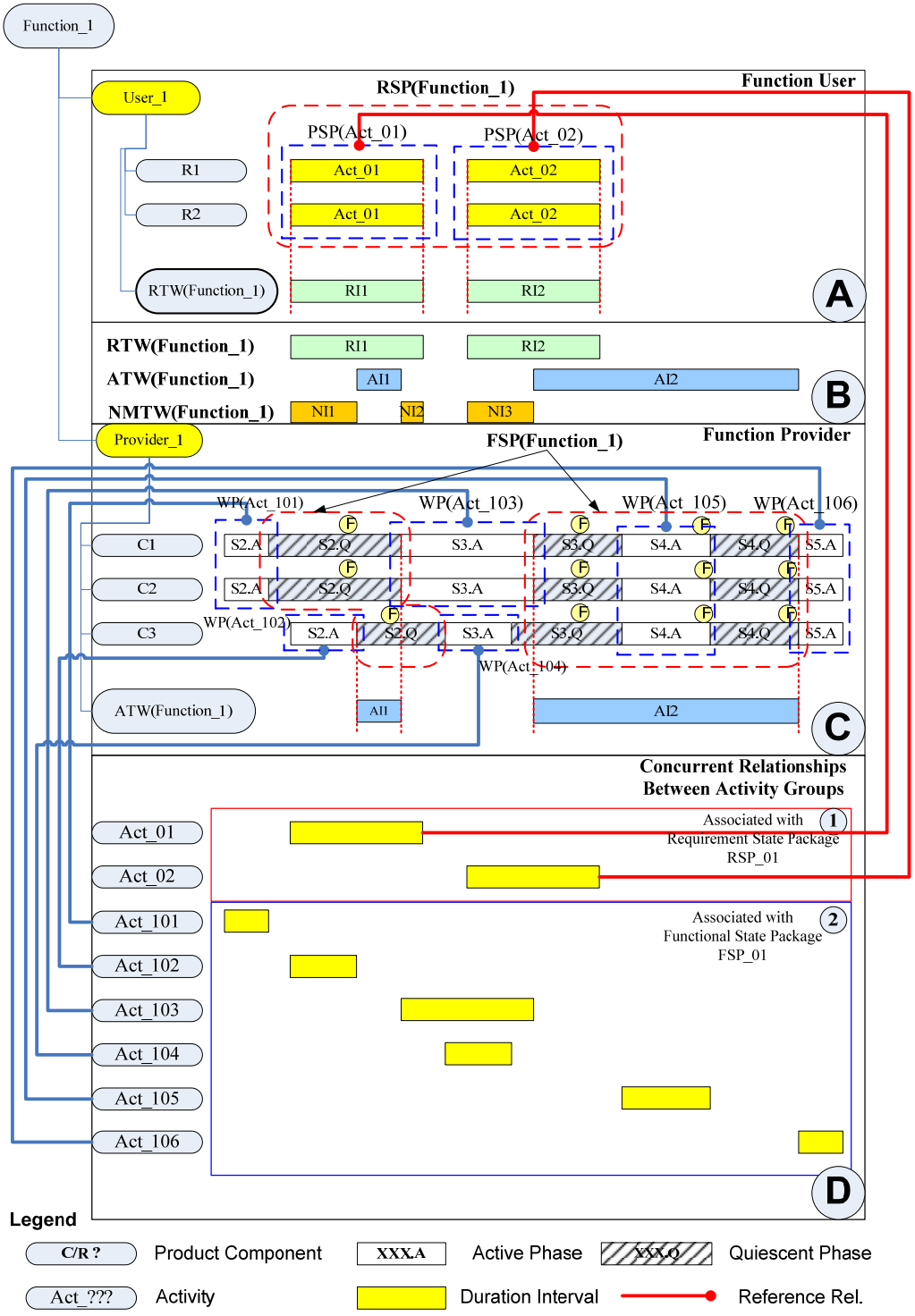



**Figure 5.1 Matching Requirement with Availability Time-Windows**

The resultant requirement time-window can contain several discontinuous intervals. For the example shown in Figure 5.1B, the requirement time window RTW(Function_1) contains two time intervals: “RI1” and “RI2”. These two intervals correspond to the duration intervals of the performer states “Act_01” (activity

"Act_01") and "Act_02" (activity "Act_02"), respectively, between which there is a quiescent gap, which makes the requirement time-window of "Function_01" discontinuous.

### 5.1.2 Computation of Availability Time-Window

The *availability time-window* of an intermediate function defines the time-window during which the intermediate functionality will be provided by the function provider, and this time-window can be derived from the associated functional states (or state phases). Only when all the constituent components in a function provider are concurrently in their functional states, can they jointly behave as an interrelated system for realizing the intermediate functionality. The computation of the *availability time-window* can be achieved in two steps. The duration intervals of the functional states of each provider component are first united into a *functional time-window* using the Boolean union operation. Then the *availability time-window* can be produced using the Boolean intersection operation on the *functional time-windows* of all provider components. The computation formula is presented as:

$$\mathbf{ATW(F)} = \bigcap_{\mathbf{i}} \left( \bigcup_{\mathbf{j}} \mathbf{I(C_i.S_j)} \right) \quad \forall\, \mathbf{C_i.S_j} \in \mathbf{FSP(F)} \qquad \mathbf{(5.2)}$$

Where $F$ is an arbitrary intermediate function, $ATW(F)$ the availability time window of $F$, $C_i.S_j$ the $S_j$ state of the $C_i$ component, $I(C_i.S_j)$ the duration interval of $C_i.S_j$, and $FSP(F)$ the functional state package of $F$.

Figure 5.1C shows the function provider "Provider_1" of "Function_1", comprising three components "C1", "C2" and "C3" along with their functional states "S2.Q", "S3.Q" and "S4" contained in the functional state package "FSP(Functional_1)". The functional state phases "S2.Q" and "S3.Q" of each

provider component indicate that the intermediate functionality is not available during its “S3.A” active phases. For the instance shown in Figure 5.1C, the availability time-window “ATW(Function_1)” is given by the following Boolean operations:

$$\begin{aligned} &\textbf{ATW(Function\_1)} \\ =\ &\textbf{((I(C1.S2.Q)} \cup \textbf{I(C1.S3.Q)} \cup \textbf{I(C1.S4)))} \\ \cap\ &\textbf{((I(C2.S2.Q)} \cup \textbf{I(C2.S3.Q)} \cup \textbf{I(C2.S4)))} \\ \cap\ &\textbf{((I(C3.S2.Q)} \cup \textbf{I(C3.S3.Q)} \cup \textbf{I(C3.S4)))} \\ =\ &\textbf{AI1} \cup \textbf{AI2} \end{aligned}$$

Figure 5.1C shows that the resultant availability time-window comprises two discontinuous time intervals, namely “AI1” and “AI2”. In this instance, it is evident from Figure 5.1C that the available time-window “ATW(Function_1)” is disrupted by the “S3.A” active phases, thus making the availability time-window discontinuous, which is often experienced in practice.

### 5.1.3 Analysis on Matching Requirement and Availability Time-windows

The temporal interaction between the user and provider of an intermediate function can be evaluated by matching the requirement time window with the availability time-window of an intermediate function. This analysis can be achieved using a Boolean cut operation between the requirement time-window (like minuend) and the availability time-window (like subtrahend). The Boolean cut operation produces the difference between two time-window operands, indicating the interval or intervals where the requirement time-window cannot be overlapped/covered by the availability time-window. The formula for computing a non-matching time-window (NMTW) is presented as:

$$\textbf{NMTW(F)} = \textbf{RTW(F)} - \textbf{ATW(F)} \qquad \textbf{(5.3)}$$

Where *F* is an arbitrary intermediate function, *NMTW(F)* the no-matching time-window of *F*, *RTW(F)* the requirement time window of *F*, and *ATW(F)* the availability time window of *F*.

The temporal interval(s) in the resultant non-matching time-window indicate(s) the periods during which the intermediate functionality is required but not provided, implying unavailable temporal interaction between the function user and the corresponding provider. Thus, an unfulfilled intermediate function can be detected. On the other hand, a "null" non-matching time-window indicates that the requirement time-window is totally covered by the availability time-window, meaning that the temporal interaction between the function user and function provider is feasible.

Figure 5.1B demonstrates the non-matching time-window NMTW(Function_1) between RTW(Function_1) and ATW(Function_1), which can derived by the Boolean cut operation as follows:

**NMTW (Function_1)**

**= RTW(Function_1) - ATW(Function_1)**

**= NI1 ∪ NI2 ∪ NI3**

Since the RTW and ATW of an intermediate functionality may not be continuous, the Boolean cut resultant can also be discontinuous. In this instance, the resultant non-matching time-window NMTW(Function_1) comprises three temporal intervals "NI1", "NI2", and "NI3" (Figure 5.1B), indicating the periods during which the "Function_1" intermediate functionality is required but cannot be provided, leading to unavailable temporal interaction between "User_1" and "Provider_1".

In this way, not only can the availability of the temporal interaction between the user and the provider of an intermediate function be evaluated, the unavailable period can also be detected and explained with respect to the associated

requirement/functional states. This information may help planning engineers improve their schedules by adjusting the schedules of the corresponding activities in order to change the component states associated with the requirement time window and the availability time-window of the unfulfilled intermediate function. Such adjustment may eliminate the non-matching time-window.

### 5.1.4 Concurrency Relationships Implied by Matching RTW with ATW

From the process modeling viewpoint, the temporal matching between the requirement time-window and the corresponding availability time-window can be explained as the concurrency relationship between the construction activities associated with the requirement states and the functional states. Also, the joint contribution of the functional states to realize an intermediate functionality implies the concurrent relationships among the associated construction activities that transit the functional states.

Figure 5.1 shows that two activities “Act_01” and “Act_02” are associated with the four requirement states of two construction performers “R1” and “R2” through two performer state packages “PSP(Act_01)” and “PSP(Act_02)”, respectively. Similarly, there are six other activities “Act_101” to “Act_106” associated with the nine functional states of three provider components “C1”, “C2” and “C3” through six work packages “WP(Act_101)” to “WP(Act_106)”, respectively. Specifically, activity “Act_106” references three “S5.A” active phases through the work package “WP(Act_106)” (Figure 5.1C). Although these three “S5.A” phases are not functional states, their start points determine the finish points of the corresponding preceding states “S4”, which are functional states.

The matching between RTW(Function_1) and RTW(Function_1) implies the concurrency relationship between the two-activity group associated with “User_1” (“Act_01” and “Act_02”) and the six-activity group associated with “Provider_1” (“Act_101” to “Act_106”). However, such concurrent relationships are not semantically represented in CPM schedules, giving rise to potential schedule conflicts because of unfulfilled intermediate function requirements.

The computation of ATW(Function_1) indicates the concurrency relationships among the six activities “Act_101” to “Act_106” (Figure 5.1D), and these concurrency relationships should be considered as temporal constraints for scheduling these six activities to fulfillment the requirement from “Function_1”. Unfortunately, such concurrency relationships determining an availability time-window are often inadequately represented in a semantic manner in traditional CPM schedules. The computation of ATW(Function_1) also implies that the concurrency relationships associated with a function provider would be better described by the quiescent gaps between the corresponding activities (like the quiescent phases “S2.Q”, “S3.Q” and “S4.Q”). However, these quiescent phases are neither depicted nor derived in the traditional CPM schedules.

## 5.2 Evaluation of Spatial Interaction between User and Provider

The matching between the available time-window and the requirement time-window only evaluates whether an intermediate functionality is available when it is required, but such evaluation does not verify whether the function user can physically interact with the provider when the temporal interaction is available. Such physical interaction can be evaluated with respect to the time-dependent spatial (or spatio-temporal) relationships between the user components and provider components. For

this purpose, the concepts of temporal space entity and temporal topological relationships between two temporal space entities are first defined. Based on these two concepts, a spatio-temporal interaction matrix can be derived to describe the time-dependent spatial interaction between the user and the provider, and then the spatio-temporal interaction matrix is evaluated against the corresponding spatio-temporal criterion matrix, which specifies the spatial interaction criteria between user components and provider components. Consequently, the unavailable spatial interaction between the user and the provider of an intermediate function can be detected.

### 5.2.1 Temporal Space Entity and Temporal Topological Relationship

The spatial interaction of an intermediate function should be evaluated from both spatial and temporal attributes of its user and its provider because the spatial interaction is not only determined by the topological relationships between the user components and the provider components, but it is also restricted by the co-existence of the associated components. However, a 3D space model that is developed using various CAD tools itself does not contain any temporal information to depict when a space entity enter or exit a space system. In such a time-independent space model, the topological relationships between the associated space entities are also time-independent. These time-independent space entities and topological relationships alone cannot be adopted to evaluate the time-dependent spatial interaction between the user and the provider. Therefore, the present study develops two concepts, namely temporal space entity and temporal topological relationship, to describe the spatial interaction between the function user and the function provider.

A temporal space entity describes both geometric characteristic and existence period of a space entity. The geometric characteristic attribute describes such spatial information as shape and location of an in-progress component, while the existence period attribute describes a time-window during which the in-progress component has the shape and stays in the location as the associated geometric characteristic depicts. For example, when an access road has been backfilled, its shape is different from when it has been excavated. So the shape of the backfilled road is different from the shape of the excavated road, and these two shapes have different existence periods.

Since a space entity may be referenced by several discontinuous functional/requirement states its existence period can contain several discontinuous intervals. For example, "backfilled" state of an access road can reference the same shape entity "R" in the CAD model as its state space and the "unexcavated" or "Original" state of the access road can also reference the same "R" space entity. This means that the space entity "R" is referenced by two states, which are discontinuous. Mathematically, the existence period of an arbitrary temporal space entity "$TSE_m$" can be derived by a Boolean union of the duration intervals of the associated requirement/functional states (or state phases) that reference "$TSE_m$" as their state spaces, as follows:

$$\mathbf{EP(TSE_m)} = \bigcup_{\mathbf{i,\,j}} \left( \mathbf{I(C_i.S_j)} \right) \qquad \forall\ \mathbf{C_i.S_j.} \in \mathbf{RSP(F)\ or\ C_i.S_j} \in \mathbf{FSP(F),}$$

$$\mathbf{IsAssociatedWith(TSE_m,\ C_i.S_j) = True} \qquad \mathbf{(5.4)}$$

Where $TSE_m$ is an arbitrary temporal space entity associated with intermediate function $F$, $C_i.S_j$ the $S_j$ state of the $C_i$ component, $I(C_i.S_j)$ the duration interval of $C_i.S_j$. $C_i.S_j \in RSP(F)$ indicates the component state $C_i.S_j$ is a requirement state contained in the requirement state package of the intermediate function $F$, while $C_i.S_j \in$ FSP(F) is a functional state contained in the functional state package of the

intermediate function $F$. $IsAssociatedWith(TSE_m, C_i.S_j)=True$ indicates that the component state $C_i.S_j$ is associated with the temporal space entity $TSE_m$ if the Boolean result is true.

Eq. (5.4) indicates that the existence period of the temporal space entity $TSE_m$ is Boolean union of all the requirement/functional states (or state phases) that reference $TSE_m$ as their state spaces. Since the duration intervals of these associated states may not be continuous, the resultant existence period may be a discontinuous time-window containing several intervals.

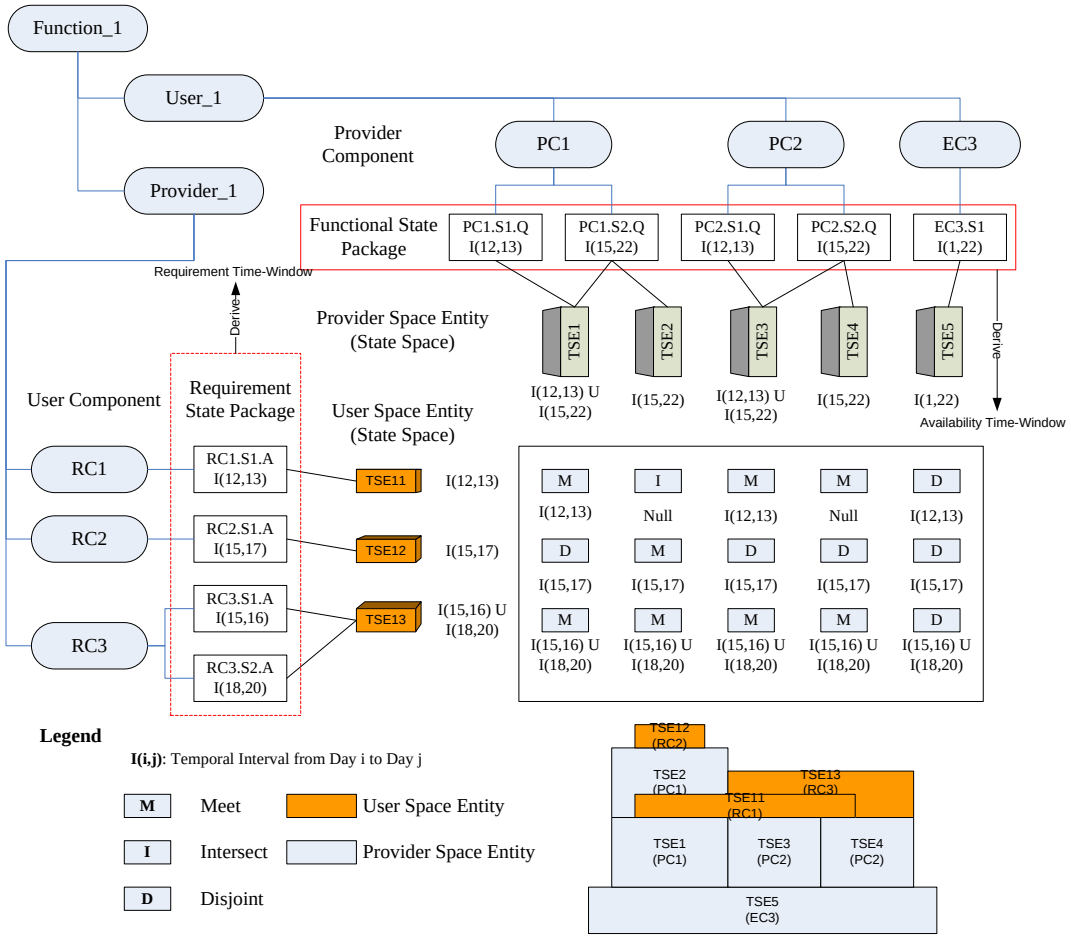



**Figure 5.2 Spatio-temporal Interaction Matrix**

Figure 5.2 illustrates the spatial interaction of a temporary support function. Specifically, two platforms “PC1” and “PC2” are erected on the ground earth “EC3”, and these three provider components work together to realize the temporary support,

while three user components, namely labor “RC1”, “RC2”, and “RC3”, requires the temporary support for executing their construction processes. According to the construction plan, “PC1” is erected in two steps which are respectively represented by its states “PC1.S1” and “PC1.S2”. The “TSE1” part of “PC1” is first erected during “PC1.S1”, and then its “TSE2” part is erected on the top of “TSE1” during “PC1.S2”. Likewise, “PC2” is also erected in two steps. Its “TSE3” part is first erected adjacent to “TSE2” during “PC2.S1”, and then the horizontally expended part “TSE4” is erected adjacent to “TSE3”.

Accordingly, there are a total of three space entities (“TSE11”, “TSE12”, and “TSE13”) associated with three user components (labor “RC1”, “RC2”, and “RC3”) via four requirement states (“RC1.S1.A”, “RC2.S1.A”, “RC3.S1.A”, and “RC3.S2.A”), while there are a total of five space entities (“TSE1” to “TSE5”) associated with three provider components (platforms “PC1” and “PC2” and earth component “EC3”) via five functional states (“PC1.S1.Q”, “PC1.S2.Q”, “PC2.S1.Q”, “PC2.S2.Q”, and “EC3.S1”) (Figure 5.2).

For example, labor “RC3” (user component) has two requirement states which reference the same temporal space entity “TSE13”, which describes the workspace of “RC3” when it executes two construction activities represented by the active phases “RC3.S1.A” and “RC3.S2.A”, respectively. Therefore, the existence period of “TSE13” can be derived as follows:

$$\mathbf{EP(TSE13) = I(RC3.S1.A) \cup I(RC3.S2.A) = I(15, 16) \cup I(18, 20)}$$

The resultant discontinuous window contains two temporal intervals during which labor “RC3” requires the temporary support functionality for executing the construction processes represented by “RC3.S1.A” and “RC3.S2.A”. Likewise, the existence periods of the other seven temporal space entities can be derived.

The temporal topological relationship between two temporal space entities represents the spatio-temporal relationships between them. A temporal topological relationship has two attributes: (binary) topological relationship and existence period. The former denotes the time-independent binary topological relationship (like disjoint, meet, and intersect) between the geometric boundaries of the two associated temporal space entities, while the latter indicates the time-window during which both associated temporal space entities co-exist to ensure the existence of the spatial interaction between them. Accordingly, the present study defines three categories of temporal topological relationships: *temporal disjoint*, *temporal meet*, and *temporal intersect*.

Mathematically, the existence period of a temporal topological relationship can be derived by Boolean intersection between the existence periods of two associated temporal space entities, given by:

$$\mathbf{EP(TTR(TSE_x, TSE_y)) = EP(TSE_x) \cap EP(TSE_{y)}} \quad \mathbf{(5.5)}$$

Where $TSE_x$ and $TSE_y$ are two arbitrary temporal space entities, and $EP(TSE_x)$ and $EP(TSE_{y)}$ are the existence periods of $TSE_x$ and $TSE_y$, respectively.

Since the existence periods of each associated space entity (like "$TSE_x$" or "$TSE_y$") may be discontinuous, the existence period of the resultant temporal topological relationship (like "$TTR(TSE_x, TSE_y)$**"**) may also be discontinuous. For instance, the existence period of the temporal topological "TTR(TSE13, TSE3)", which represents the spatio-temporal relationship between the user space entity "TSE13"and the provider space entity "TSE3" (Figure 5.2), can be derived as follows:

**EP(TTR(TSE13, TSE3))**

**= EP(TSE13) ∩ EP(TSE3)**

**= (I(15, 16) ∪ I(18, 20)) ∩ (I(12, 13) ∪ I(15, 22))**

**= I(15, 16) ∪ I(18, 20)**

Incidentally, this discontinuous existence period is equal to the existence period of the associated user space entity "TSE13" (derived in the preceding paragraph). This means that when labor "RC3" requires the temporary support (during I(15,16) and I(18,20)) from the platform "PC2", the former can "Meet" the latter.

### 5.2.2 Analysis of Spatio-Temporal Interaction Matrix using Spatio-Temporal Criterion Matrix

Figure 5.2 illustrates the fifteen temporal topological relationships between 3 user space entities and 5 provider space entities organized into a 3*5 matrix, called a spatio-temporal interaction matrix. A spatio-temporal interaction matrix of an intermediate function depicts the spatial interaction between its user and provider component with respect to the temporal topological relationships between the temporal space entities associated with each requirement state and each functional state. In such a matrix, the space entity at each row represents a user space entity, while the space entity at each column represents a provider space entity.

Figure 5.2, for example, shows that "TSE13" at the third row is referenced by the user component "RC3" via the requirement state phases "RC3.S1.A" and "RC3.S2.A", while "TSE3" at the third column is reference by the provider component "PC2" via the functional state phases "PC2.S1.Q" and "PC2.S2.Q". Each element in the interaction matrix denotes the temporal topological relationship between the user (row) space entity and provider (column) space entity. For instance,

the element at the third row and the third column depicts the temporal topological relationship between the temporal space entities “TSE13” and “TSE3”. In this way, the spatial interaction between the function user and the corresponding provider can be described by the corresponding spatio-temporal interaction matrix.

The derived spatio-temporal interaction matrix is validated against the corresponding spatio-temporal criterion matrix in order to detect unavailable spatial interaction. The spatio-temporal criterion matrix defines the expected spatio-temporal relationships between the user components and provider components with respect to the necessary conditions/criteria for the topological relationships between the user (row) space entities and provider (column) space entities. Thus, the size of a spatio-temporal criterion matrix should be equal to that of the corresponding spatio-temporal interaction matrix. Moreover, the sequences of space entities in the row and the column are also the same as the corresponding spatio-temporal interaction matrix.

For example, Figure 5.3 shows a 3*5 spatio-temporal criterion matrix for evaluating the spatio-temporal interaction matrix illustrated in Figure 5.2. Specifically, “TSE3” is the third row entity in both the interaction matrix and the criterion matrix (see Figures 5.2 and 5.3). In this way, each element in the spatio-temporal interaction matrix can be matched with the criterion element at the same row and column in the spatio-temporal matrix. For instance, the temporal topological relationship TTR(TSE13, TSE3) at the third row and the third column in interaction matrix shown in Figure 5.2 can evaluated according to the criterion defined in the third row and the third column in the criterion matrix illustrated in Figure 5.3.

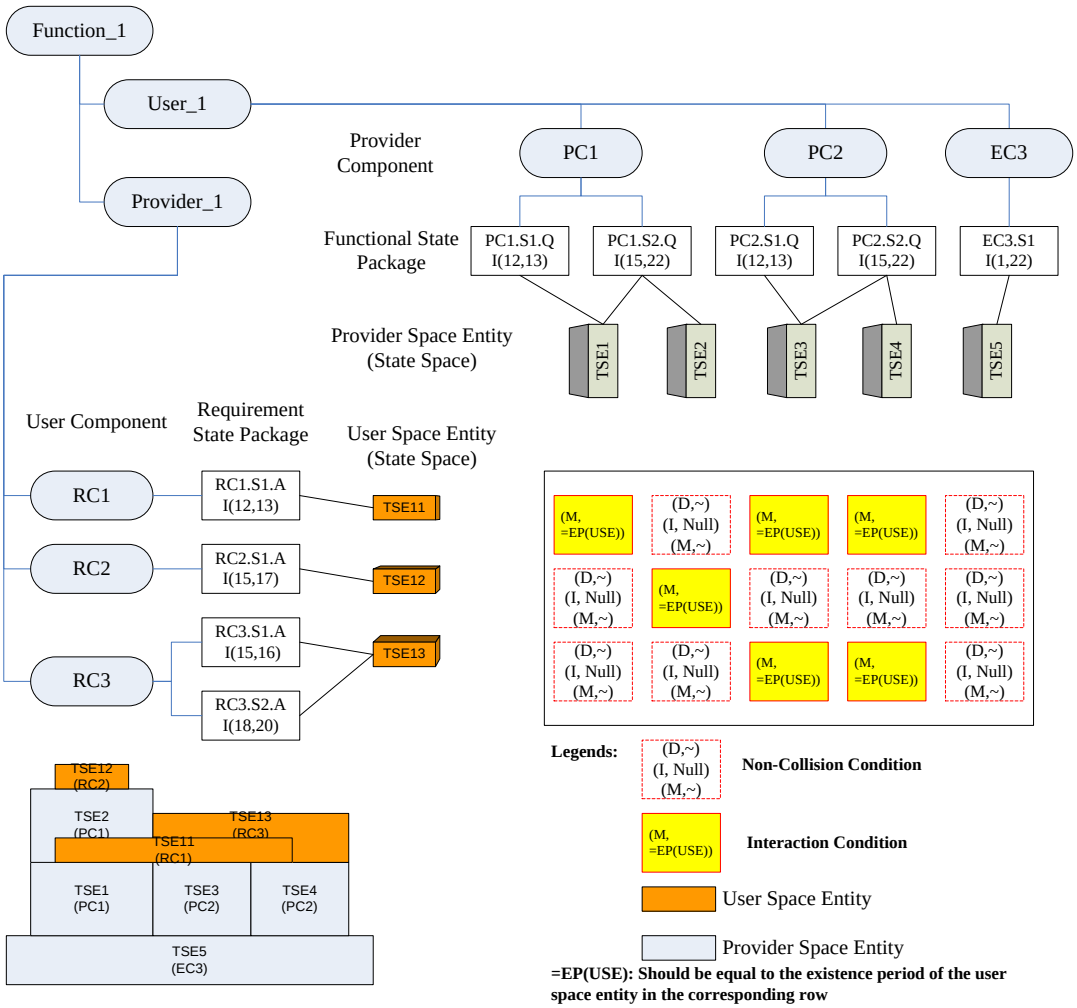



**Figure 5.3 Spatio-Temporal Criterion Matrix**

There are two types of conditions in a spatio-temporal criterion matrix. The first is the “interaction” condition, while the second is the “non-collision” condition. The “interaction” condition is necessary for realizing the required spatial interaction between the user and provider components. Such a necessary condition defines how and when the associated user components should spatially interact with the associated provider components with respect to the required type(s) of temporal topological relationship and the required range for its existence period. On the other hand, the “non-collision” condition defines that temporal topological relationship between the user space entity and the corresponding space entity does not affect the spatial

interaction between the user and the provider as long as they do not collide with each other.

In the context of evaluating temporary support functionality, an interaction element only permits a temporal meet relationship, and its existence period should be equal to the existence period of the associated user space entity, indicating that the associated user component can temporally access the associated provider component whenever the user component requires the temporary support. Since the existence period of a temporal topological relationship is the Boolean intersection of the existence periods of the associated user and provider space entities, the existence period of the resultant temporal topological relationship can never be longer than the existence period of either user or provider space entity. Thus, the “interaction” condition for a temporary support function is denoted by “(M, =EP(USE))”, where “M” denotes that the spatio-temporal relationships between the user and provider space entities should be temporal meet and “=EP(USE)” denotes that the existence period of the corresponding temporal topological relationships should be equal to the associated user space entity.

On the other hand, a “non-collision” condition for a temporary support function does not permit a temporal intersection relationship with non-zero existence period, which would indicate a temporal collision between the associated user space entity and the associated provider space entity. This condition is therefore denoted by “(I, null)”. Meanwhile, a temporal disjoint relationship or a temporal meet relationship with any existence period is permitted for a “non-collision” condition without causing any spatial conflict. These two criteria are denoted by “(D, ~)” and “(M, ~)”, respectively.

The example in Figure 5.3 shows that a total of six elements in the spatio-temporal criterion matrix are assigned the “interaction” condition, depicting the necessary conditions to realize the spatial interaction between “User_1” and “Provider_1”. Herein, C(TTR($TSE_u$, $TSE_p$)) denotes the condition for the temporal topological relationship between the user space entity $TSE_u$ and the provider space entity $TSE_p$. Specifically, with respect to “TSE11”, “C(TTR(TSE11, TSE1))”, “C(TTR(TSE11, TSE3))”, and “C(TTR(TSE11, TSE4))” denotes that the workspace of labor “RC1” (“TSE11”) should “meet” the lower part of the platform “PC1” (“TSE1”), the horizontally expanded “PC2” (“TSE3”, and “TSE4”) for accessing the temporary support. Similarly, “C(TTR(TSE12, TSE2))” defines that the work space of “RC2” (“TSE12”) should meet the elevated part (“TSE2”) of the platform “PC1”.

Figure 5.3 also illustrates that a total of nine elements in the spatio-temporal criterion matrix have assigned the “non-collision” condition. Specifically, with respect to “TSE5”, “C(TTR(TSE11, TSE5))”, “C(TSE12, TSE5))”, and “C(TSE13, TSE5))” denotes that the labor “RC1”, “RC2” and “RC3” has no need to “meet” the earth component “EC3” under two platforms “PC1” and “PC2”. Similarly, “C(TTR(TSE11, TSE2))” indicates that the labor “RC1” (whose workspace is “TSE11”) does not require but should avoid temporal collision with the elevated part (physical space “TSE2”) of the platform “PC1”, and “C(TTR(TSE13, TSE2))” means that the “RC2” in its workspace “TSE13” does not require the temporary support from the lower part(“TSE2”) of the platform “PC1” as long as these “TSE13” and “TSE2” do not collide with each other. The section view in Figure 5.3 shows that the user space entity “TSE13” (the workspace of “RC3”) actually meet “TSE2” (physical space of the elevated part of “PC1”) on their vertical sides.

The unavailable spatial interaction between the function user and the corresponding function provider can be detected by evaluating the spatio-temporal interaction matrix against the corresponding spatio-temporal criterion matrix. If any temporal topological relationship in the interaction matrix cannot fulfill the condition specified in the corresponding element in the spatio-temporal criterion matrix, an undesirable temporal topological relationship is detected, implying unavailable spatial interaction. An undesirable temporal topological relationship can arise from either prohibited types of spatio-temporal relationship occurs or its existence period cannot satisfy the range defined in the spatio-temporal criterion matrix.

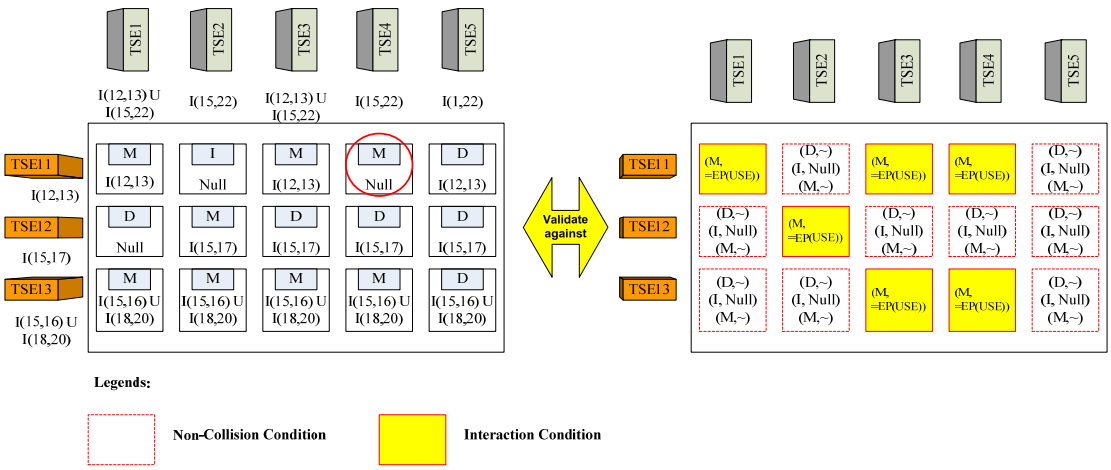



**Figure 5.4 Matching Interaction Matrix with Criterion Matrix**

Figure 5.4, for example, illustrates the validation of the spatio-temporal interaction matrix shown in Figure 5.2 against the spatio-temporal criterion matrix defined in Figure 5.3. This validation detects one undesirable temporal topological relationship TTR(TSE11, TSE4) at the first row and the fourth column of the spatio-temporal interaction matrix since its existence period is null instead of being equal to the existence period of the corresponding user space entity “TSE11”, indicating that The “interaction” condition “M, =EP(USE)” for TTR(TSE11, TSE4) is violated. This means that labor “RC1” cannot access the horizontally expanded part of platform “PC2” on Days 12 and 13. Additionally, Figure 5.4 shows that although “TSE11”

intersects with “TSE2”, but the existence period of this temporal intersection relationship is “null”. Therefore, this will not result in temporal collision.

Consequently, the analysis of evaluating spatio-temporal interaction matrix against spatio-temporal criterion matrix not only detects the inaccessibility to the function provider, but the analysis result also indicates the undesirable temporal topological relationships in terms of undesirable temporal topological types or improper existence periods. This information can aid planning engineers in adjusting construction schedules for resolving the unavailable spatial interactions.

### 5.2.3 Example of Moving Mobile Crane on Excavated Access Road

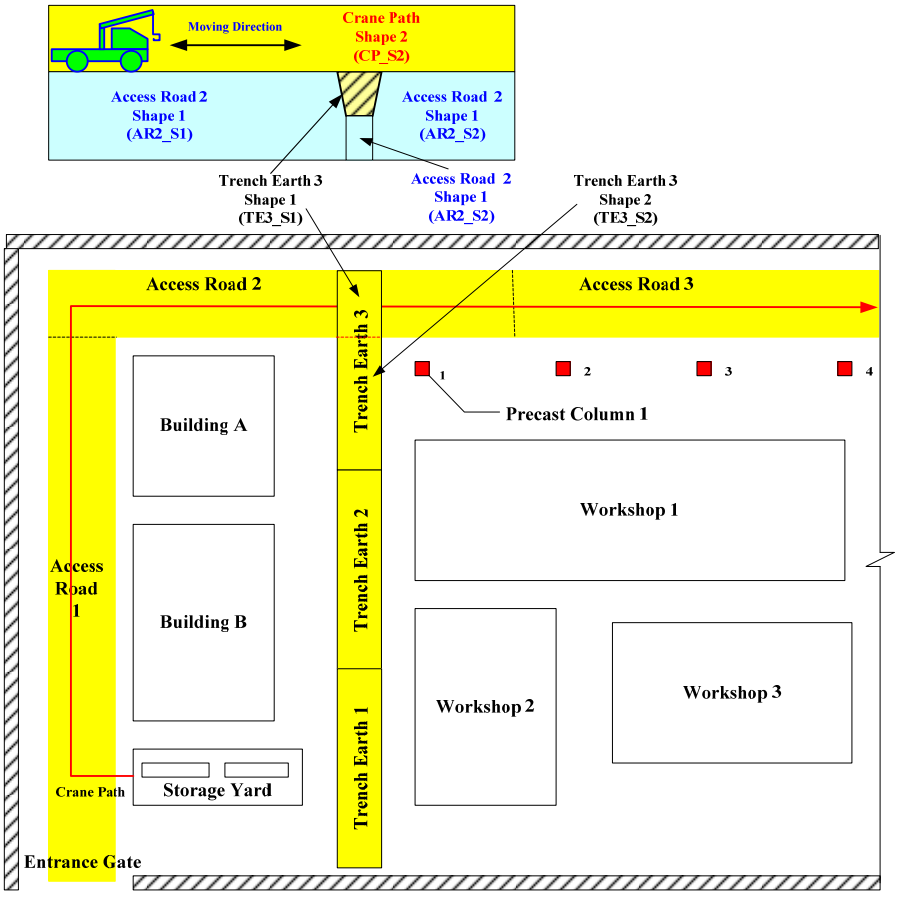



**Figure 5.5 Site Layout for Moving Mobile Crane**

Figure 5.5 illustrates an example, using a 2D drawing, that a mobile crane is employed to transport precast columns from the storage yard to the erection positions

and then to lift and install the columns. This example is used to explain the detection of unavailable spatial interaction by evaluating the spatio-temporal interaction matrix against the corresponding spatio-temporal criterion matrix. At the same time, the site is also undergoing underground piping work that takes place across the site as shown in Figure 5.5. This construction scenario implies that inaccessibility problems may occur as a result of the excavation of the access road.

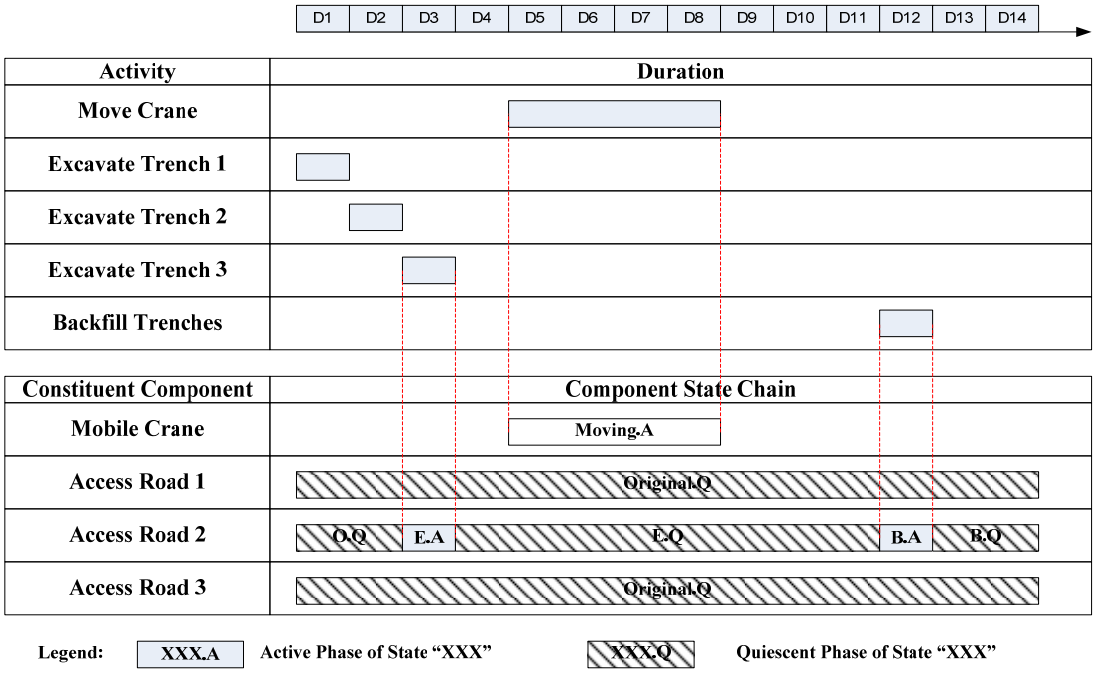



**Figure 5.6 Durations of Related Activities and Component States**

The above-mentioned access scenario is depicted with respect to the process activities and the associated product/resource components as well as their construction life cycles. It is also assumed that the period of the study is 14 working days (Day 1 to Day 14). Figure 5.6 shows five construction activities whose start and finish times are presented for deriving the durations of the corresponding state phase. In particular, "Move Crane" is a logistic activity that will be executed from the start of Day 5 to the end of Day 8. The access road on which the crane is moving is divided into three segments labeled as "Access Road 1", "Access Road 2", and "Access Road 3" (see Figure 5.5). The trench work will be sequentially executed in three zones, labeled as

“Trench Earth 1”, “Trench Earth 2”, and “Trench Earth 3” on Days 1, Day 2, and Day 3, respectively. The excavation of “Trench Earth 3” will change the shape of the “Access Road 2”. After the pipe work finishes, the three excavated trenches will be concurrently backfilled on Day 12.

Figure 5.6 also illustrates the association relationships between the activities and the corresponding component states. Specifically, the “Move Crane” activity is mapped onto the “Mobile Crane” as its performance state phase “Moving.A”, whose duration is from the start of Day 5 to the end of Day 8. Two construction activities “Excavate Trench 2” and “Backfill Trenches” are respectively mapped as the active phases of the “Excavated” and “Backfilled” states of “Access Road 2”. Accordingly, the active phases of these two active state phases are on Day 3 and Day 12, respectively. Meanwhile, the quiescent state phases of the relevant components are also shown in Figure 5.6. Specifically, the “Excavated.Q” quiescent phase of “Access Road 2” lasts from the start of Day 4 to the end of Day 11. Both “Access Road 1” and “Access Road 3” respectively have only one quiescent state “Original.Q”, which lasts from the beginning to end of the 14-day study period, since no activity is executed on them.

Figure 5.7 shows the semantic model of the intermediate function “Temporary Support Function”, depicting the functional interdependency between the “Mobile Crane” and the “Access Road”. The “User” comprises only one construction performer “Mobile Crane”, while the function provider comprises three road segments “Access Roads 1 to 3”. Accordingly, the requirement state package comprises only one active phase, i.e. “Moving.A” of “Mobile Crane”, while the functional state package comprises five state phases associated with three road segments, respectively. The functional state phases for each road segment can be either “Original.Q”,

"Excavated.Q", or "Backfilled.Q" because the access road segments cannot provide support functionality when it is being excavated ("Excavated.A") or backfilled ("Backfilled.A").

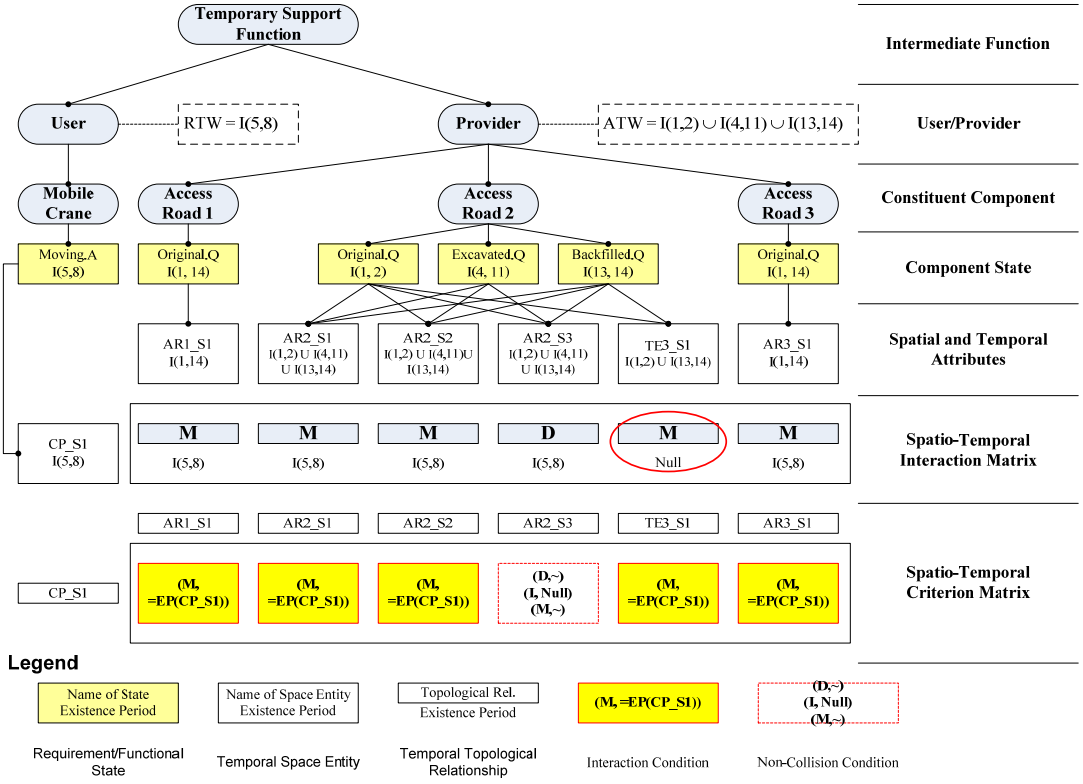



**Figure 5.7 Spatio-temporal Interaction Matrix for Detecting Inaccessibility**

Follow Eq. (5.1) and Eq. (5.2), it can be derived that the requirement time-window RTW(Temporary_Support_Function) is the time interval I(5,8), while the availability time-window ATW(Temporary_Support_Function) is the discontinuous time-window I(1,2) ∪ I(4,11) ∪ I(13,14). Following Eq. (5.3), the corresponding non-matching time-window between the RTW and the ATW is "null", indicating that the temporal interaction between "User" and "Provider" is available. This means that the support functionality can be provided by the access road whenever the "Mobile Crane" requires it. On the other hand, whether the mobile crane can access the provided support functionality should be further evaluated with respect to the spatial interaction between the crane and the access road.

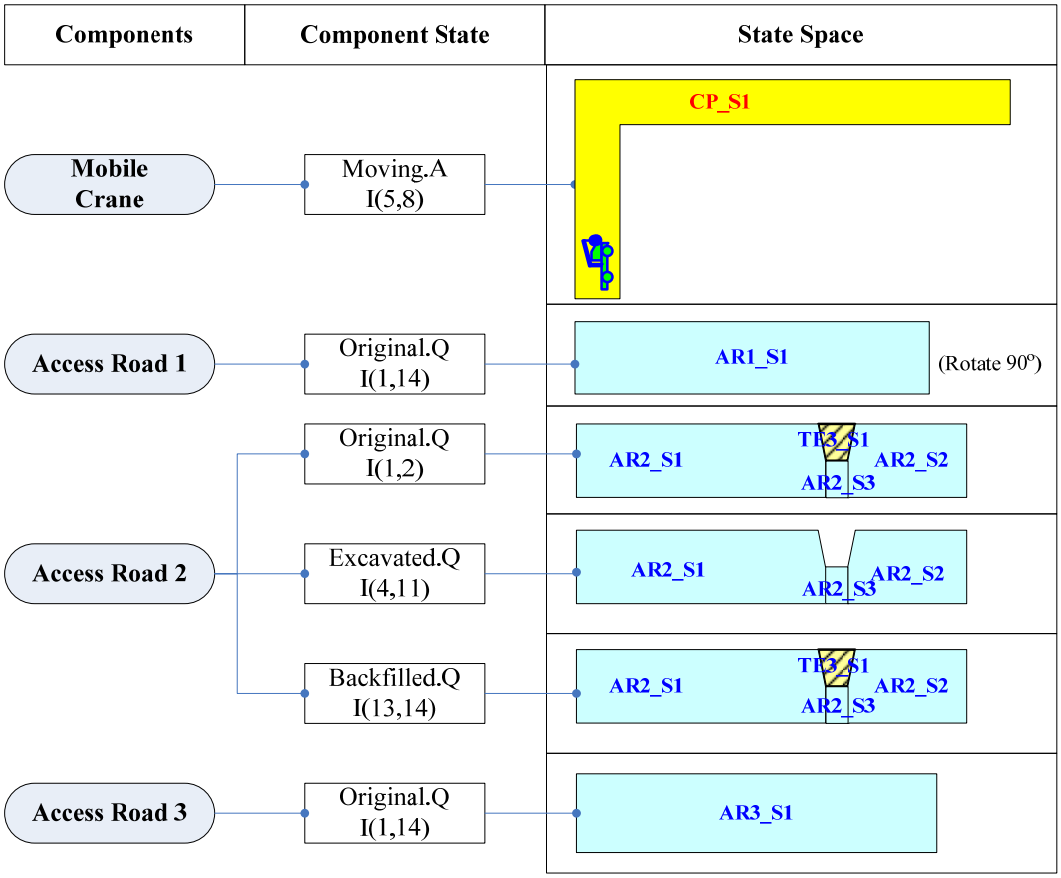



**Figure 5.8 State Space Attributes of Component States**

Figure 5.8 illustrates the space entities referenced by the state space attribute of each requirement/functional component state (phase) of "Temporary Support Function". Specifically, the state space of "Moving.A" references the path space entity "CP_S1" to represent the trajectory boundary of "Mobile Crane". On the other hand, each functional state of the three road segments also references one or several space entities to describe their physical spaces. In particular, both "Original.Q" and "Backfilled.Q" functional state phases of "Access Road 2" reference the same four physical space entities: "AR2_S1", "AR2_S2", "AR2_S3", and "TE3_S1" (see Figure 5.5), but its "Excavated.Q" phase references only three space entities since the "TE3_S1" space entity, representing the earth object of the excavated trench, is excluded from the state space of "Excavated.Q" during which the trench is kept open. Additionally, the "Original.Q" quiescent phases of the unexcavated road segments

“Access Road 1” and “Access Road 3” reference the physical space entities “AR1_S1” and “AR3_S1”, respectively.

Accordingly, the existence period of each temporal space entity shown in Figure 5.8 is the Boolean union of the duration intervals of the associated requirement/functional state phases. Specifically, the existence period of “CP_S1” (the path space entity of “Mobile Crane”) is equal to the duration interval of state phase “Moving.A” of “Mobile Crane”, i.e. I(5, 8). Similarly, the existence periods of both “AR1_S1” (“Access Road 1”) and “AR3_S3” (“Access Road 3”) encompass the whole study period, i.e. I(1,14). On the other hand, the existence period of “TE3_S1”, referenced by two functional state phases “Original.Q” and “Backfilled.Q” of “AR2_S2”, can be derived as follows:

$$\mathbf{EP(TE3\_S1) = I(AR2\_S2.Original.Q) \cup I(AR2\_S2.Backfilled.Q)}$$

$$\mathbf{= I(1,2) \cup U\ I(13,14)}$$

The existence period of the other three temporal space entities “AR2_S1”, “AR2_S2”, and “AR2_S3” can be likewise derived.

The temporal topological relationships between the path space entity (user space entity) of the mobile crane and the six physical space entities (provider space entities) associated with the three road segments are organized into an 1*6 spatio-temporal interaction matrix as shown in Figure 5.7. There are a total of five temporal meet relationships and one temporal disjoint relationship in the spatio-temporal interaction matrix.

The existence period of each temporal topological relationship in the spatio-temporal interaction matrix can be derived by the Boolean intersection of the existence periods of the associated user and provider space entities. For example, the existence period of TTR(CP_S1, TE3_S1) is derived as follows:

**EP(TTR(CP_S1, TE3_S1))**

**= EP(CP_S1) ∩ EP(TE3_S1)**

**= I(5,8) ∩ (I(1,2) ∪ I(13,14)) = Null**

The resultant "null" existence period of TTR(CP_S1, TE3_S1) indicates that the "Mobile Crane" may not temporally "meet" the trench earth object in the road segment "Access Road 2" since they cannot co-exist in the intermediate function system. The derived spatio-temporal interaction matrix will be evaluated against the spatio-temporal criterion matrix that is presented in the bottom of Figure 5.7 for detecting undesirable temporal topological relationships.

The spatio-temporal criterion matrix in Figure 5.7 denotes that the path space of the "Mobile Crane" should meet each of the six road space entities. In detail, the six elements in the spatio-temporal criterion matrix are all assigned the interaction condition (M, =EP(CP_S1)). Herein, "M" denotes that the type of each temporal topological relationship should be "temporal meet", while "=EP(CP_S1)" indicates that its existence period should be equal to the existence period of the crane path space "CP_S1".

The evaluation identifies one undesirable temporal topological relationship TTR(CP_S1, TE3_S1), which is at the fifth column. Its "null" existence period is unable to satisfy the corresponding existence period criterion "=EP(CP_S1)" in the interaction condition. This means that the moving path of "Mobile Crane" should but can not "meet" the trench earth object ("TSE3_S1") from the start of Day 5 to end of Day 8 when it has been excavated. This renders the required temporary support functionality inaccessible due to the improper construction schedule. This result is consistent with the fact that the mobile crane cannot move over the excavated trench

at the site during Days 5 to 8 as planned. Consequently, the unfulfilled intermediate function requirements arising from the infeasible spatial interactions can be detected.

## 5.3 Analysis on Matching Multiple Users with Multiple Providers

Some intermediate functionalities are compatible or substitutable on site. In this case, if the similar intermediate functionalities are requested by a number of function users and if they are themselves provided by more than one provider, the "non-matching" evaluation discussed earlier can be extended from matching single user with single provider to matching multiple users with multiple providers. For example, a worker (function user) can access the work faces from different accesses (alternative function providers), and a scaffold (function provider) can support the construction works of different trades (multiple function users). This implies that if the requirement/availability time-windows of these users and providers can be properly matched, some temporary facilities may be avoided, and the risks of inaccessibility to work faces may be reduced. Consequently, the executability of the construction schedules be improved, leading to better constructability of a facility project.

The fulfillment of a group of compatible intermediate function requirements can be evaluated with respect to matching the time-windows between the associated users and providers. The "non-matching time-window" (denoted by *mNMTW* where "m" denotes "multiple") between the requirement time-windows of multiple users (in this case, *n* users) and the availability time-windows of multiple providers (in this case, *n* providers) can be evaluated using the Boolean cut operation on the Boolean union resultants of the respective requirement and availability time-windows, given by the equation Eq. (5.6):

$$\mathbf{mNMTW}(\mathbf{F}_1,\ldots,\mathbf{F}_i,\ldots,\mathbf{F}_n) = \bigcup_{i=1}^{n}(\mathbf{RTW}(\mathbf{F}_i)) - \bigcup_{i=1}^{n}(\mathbf{ATW}(\mathbf{F}_i))$$

$$\forall\ \mathbf{i} \in \mathbf{\{1,\ldots,n\}} \qquad \mathbf{(5.6)}$$

Where $n$ intermediate functions $F_1$ to $F_n$ are compatible with each other, $F_i$ the $i^{th}$ intermediate function in the compatible intermediate function set $\{F_1, \ldots, F_i, \ldots, F_n\}$, $RTW(F_i)$ the requirement time-window of $F_i$, and $ATW(F_i)$ the availability time-window of $F_i$.

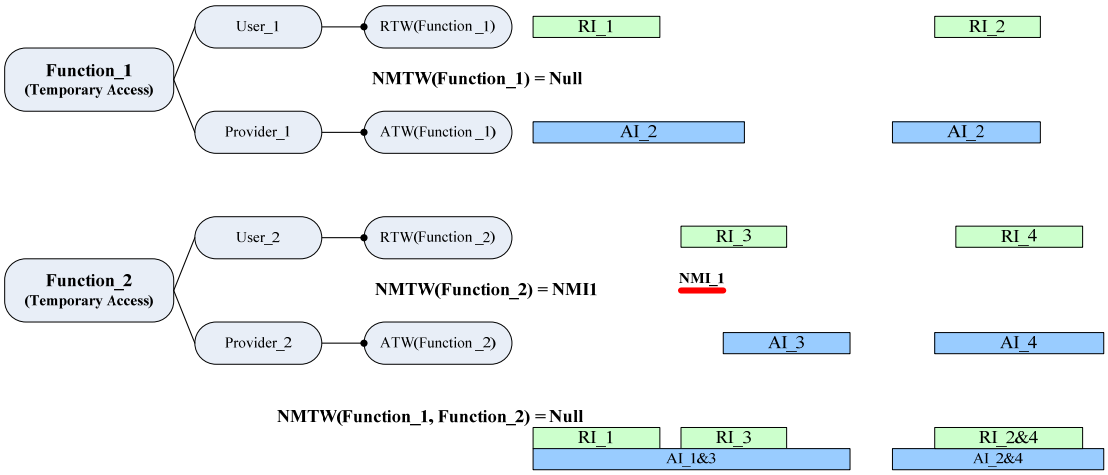



**Figure 5.9 Non-Matching Time-Window between Multiple Users and Providers**

Figure 5.9, for example, shows two users ("User_1" and "User_2") and two providers ("Provider_1" and "Provider_2") associated with two compatible intermediate functions "Function_1" and "Function_2", respectively. The requirement time-window RTW(Function_1) of "User_1" contains two intervals "RI_1" and "RI_2", while RTW(Function_2) of "User_2" contains another two intervals "RI_3" and "RI_4". The availability time-window ATW(Function_1) of "Provider_1" comprises two intervals "AI_1" and "AI_2", while ATW(Function_2) of "Provider 2" comprises "AI_3" and "AI_4".

Employing Eq. (5.3) to evaluate the single intermediate function "Function_2", the resultant non-matching time-window NMTW(Function_2) contains the interval "NMI_1", indicating that the requirement interval RI_3 cannot be completely

contained by the availability interval “AI_3” (see Figure 5.9). This means that during the interval “NMI_1”, the intermediate function requirement from “User_2” cannot be fully fulfilled by the corresponding “Provider_2”. This dilemma, however, can be resolved by the “Provider_1” of the alternative “Function_1”.

Figure 5.9 illustrates the Boolean union of two requirement time windows RTW(Function_1) and RTW(Function_2), which is a discontinuous time-window comprising three intervals (RI_1, RI_3, RI_2&4). This time-window denote the time that either “User_1” or “User_2” would require the compatible intermediate functionalities “Function_1” and “Function_2”. Note that “RI_2&4” is the Boolean union of the overlapped “RI_2” and “RI_4”. On the hand, the Boolean union of two available time windows, comprising two intervals AI_1&3 and AI_2&4, denotes the time-window during which the compatible intermediate functionalities can be realized by the two engineering alternatives provided by “Provider_1” and “Provider_2”. AI_1&3 and AI_2&4 are two continuous intervals respectively made up of two availability intervals, namely AI_1 overlapping with AI_3 and AI_2 overlapping with AI_4. The non-matching time-window of “Function_1” and “Function_2” can be derived using Eq. (5.6) as:

$$\begin{aligned}
&\textbf{mNMTW(Function-1, Function_2)} \\
&= \textbf{(RTW(Function_1)} \cup \textbf{RTW(Function_2))} \\
&\quad \textbf{- (ATW(Function_1)} \cup \textbf{ATW(Function_2))} \\
&= \textbf{(RI_1} \cup \textbf{RI_2} \cup \textbf{RI_3} \cup \textbf{RI_4) – (AI_1} \cup \textbf{AI_2} \cup \textbf{AI_3} \cup \textbf{AI_4)} \\
&= \textbf{(RI1} \cup \textbf{RI3} \cup \textbf{RI_2\&4) - (AI_1\&3} \cup \textbf{AI_2\&4)} \\
&= \textbf{Null}
\end{aligned}$$

By using the substitutable engineering solutions to resolve the compatible intermediate function requirements, the resultant mNMTW(Function_1, Function_2)

is null, indicating that unfulfilled requirement during the interval “RI_3” of “User 2” can now be achieved by the functionality provision during the interval “AI_1” of “Provider 1”. In this way, the compatible intermediate functionalities required by the two users can be collaboratively realized by both engineering solutions.

Furthermore, Eq. (5.6) also implies that adjusting a requirement interval of a function user may save expenditure on temporary facilities. Specifically, it is assumed that the function user “$U_k$” of the intermediate function “$F_k$” is originally supported by the corresponding provider “$P_k$”. If the requirement interval “$RI_x$” of “$U_k$” can be adjusted to utilize the compatible intermediate functionality provided by the provider “$P_j$” of another intermediate function “$F_j$”, some temporary facilities associated with the provider “$P_k$”, which were originally necessary during the requirement interval “$RI_x$” for supporting the user “$U_k$” need not be constructed or can be dismantled/removed earlier, leading to saving of cost.

Furthermore, Eq. (5.6) also implies that if an requirement interval “$RI_x$” of an function user “$U_k$” can be adjusted to utilize the intermediate functionality provided by other provider(s), some provider components, especially temporary facilities, that are once required for realizing the intermediate functionality for supporting the user “$U_k$” during the original “$RI_x$”, can be saved or dismantled/removed earlier, which means potential saving.

For example, Figure 5.9 shows that the requirement interval “RI_4” of “User_2” partially overlaps with the availability interval “AI_2” of “Provider_1” and that the length of “RI_4” is shorter than the length of “AI_2”. If “RI_4” can be advanced until it is contained by “AI_2” the availability interval “AI_4” of “Provider_2” may not be required. This means the provider component(s) required for realizing the “Function_2” functionality during “AI_4”, for example a scaffold, can be

avoided. This can be achieved by the collaborative planning among relevant contractors.

## 5.4 Identification of Bottleneck State

The principle of lean construction states that the delay of some construction activities, especially critical activities, arises from waiting for the availability of the associated resources and design information. Likewise, the late commencement of some construction or logistic activities may result from the late availability of intermediate functionalities for supporting the associated construction performers and maintaining the temporary stability of the associated in-progress structure. In this case, shortening a construction schedule can be realized by advancing the availability of the intermediate functionalities that determines the commencement of the associated construction activities.

In general, a functional state package may comprise a number of functional states which may not start simultaneously. A *Bottleneck state* is a functional state that determines the earliest availability of an intermediate functionality that constraints the commencement of the associated construction/logistic activity. Such an "earliest availability" time point can be either the start of an availability time-window or the start of an availability interval contained in a discontinuous availability time-window. In other words, if an activity is constrained by the first interval of an availability time-window, the bottleneck state is the functional state that determines the start point of the availability time-window. If an activity is constrained by the $i^{th}$ availability interval contained in a discontinuous availability time-window, the bottleneck state is the functional state that determines the start of the $i^{th}$ availability interval.

The definition of the bottleneck state implies that a bottleneck state can be detected by comparing the start time points of the functional states/phases that are associated with the delayed construction activity via its construction performers or its work package. Figure 5.10 illustrates a typical case of an availability time-window containing an interval "AI1" where the bottleneck state occurs. In this case, the commencement of "Activity B" has been assumed to be determined by the availability of "Function X" since its construction performers "Labor A" and "Equipment A" (user components) requires the "Function X" functionality provided by the product components "G1" and "H1" (provider components). Accordingly, the functional state package comprises the "S3.Q" and "S4" states of both "G1" and "H1".

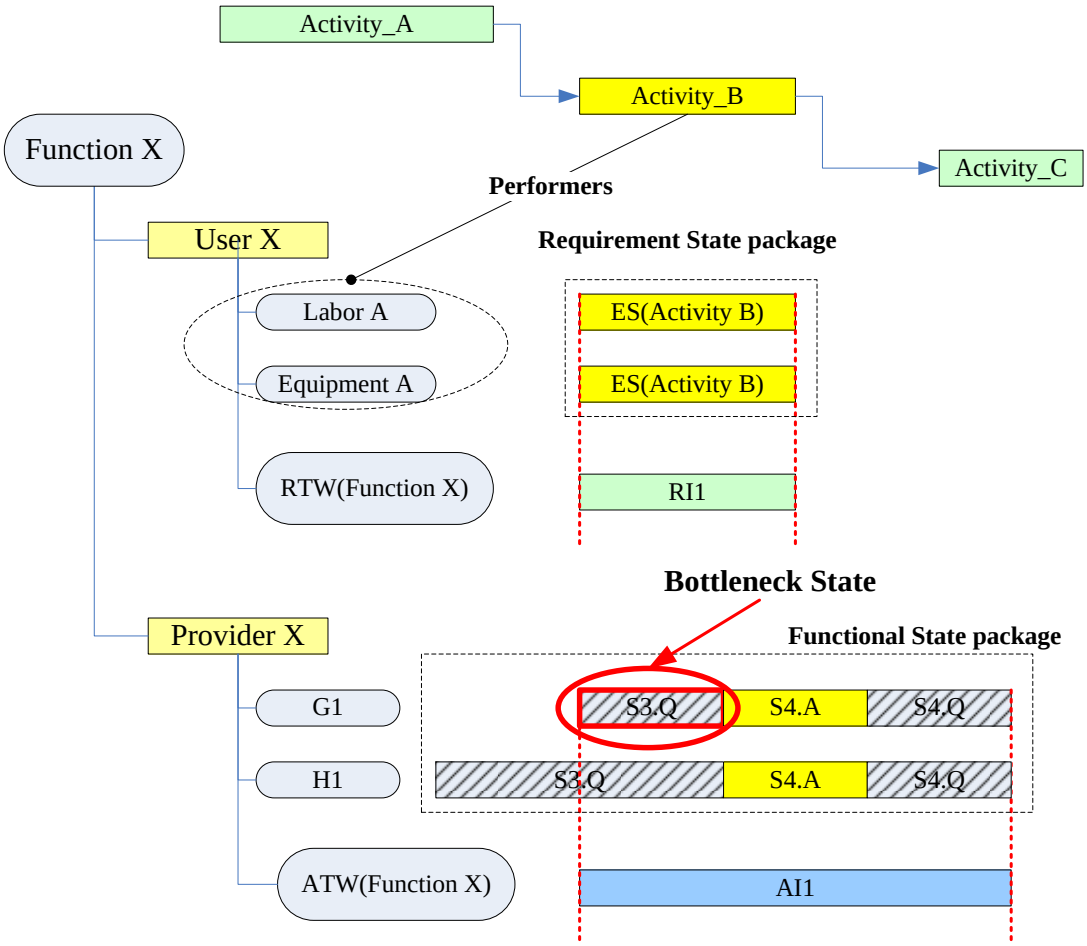



**Figure 5.10 Bottleneck States with Single-Interval Availability Time-Window**

Figure 5.10 further shows that the earliest start of a continuous availability time-window containing only one interval “AI1”, whose start is determined by the quiescent phase “S3.Q” of “G1” because it starts later than the “S3.Q” of “H1”. This means that the advancement of “S3.Q” of “G1” toward the start of the “S3.Q” of “H1” can bring forward the availability time-window of “Function X” functionality provided that each predecessor activity (like “Activity A”) of “Activity B” has a float time, leading to the earlier commencement of activity “Activity B”. Additionally, the advancement of the quiescent phase “S3.Q” implies the corresponding active phase “S3.A” should also be brought forward simultaneously. Therefore, the “S3” state of “G1” is the bottleneck state that constraints the start of “Activity B”.

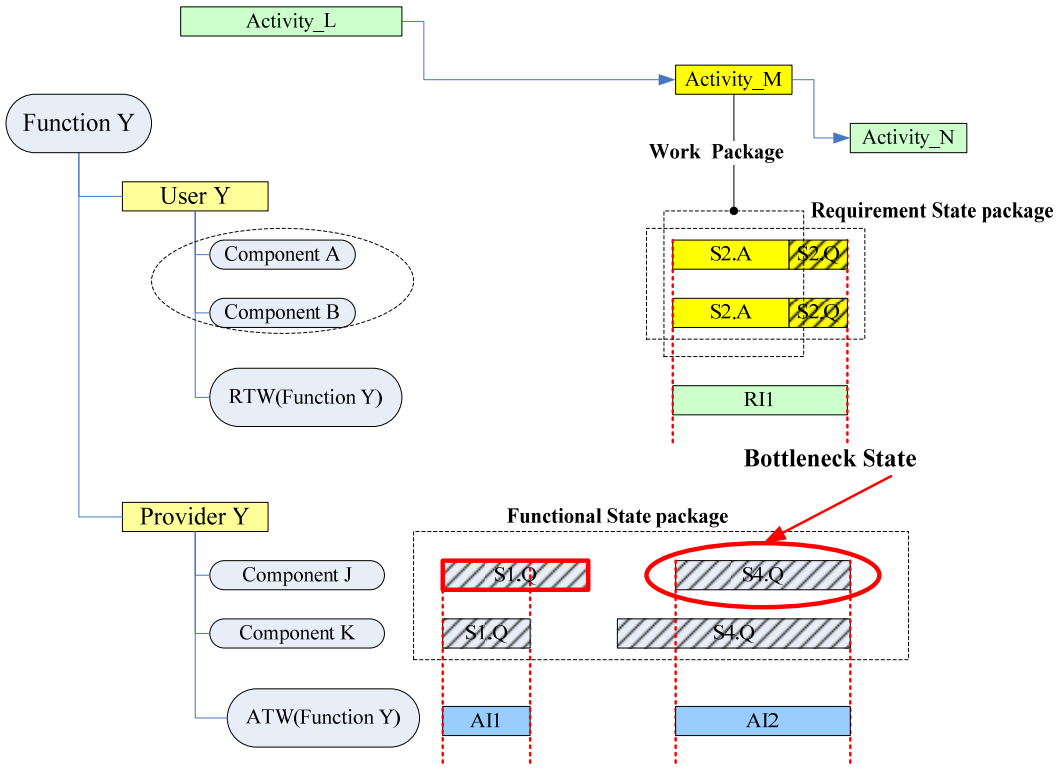



**Figure 5.11 Bottleneck States with Discontinuous Availability Time-Window**

Figure 5.11 shows a bottleneck state that occurs during the second interval of the availability time-window ATW(Function Y). The Figure also illustrates that the work package of “Activity M” comprises the “S2.A” active phases of “Components A

and B". These two in-progress components require the "Function_Y" functionality for maintaining their temporary stability. This means that the commencement of "Activity M" is constrained by the availability of the "Function_Y" functionality, which is provided by "Components J and K" in their "S1.Q" and "S4.Q" quiescent states. The availability time-window ATW(Function Y) is a discontinuous time-window containing two intervals "AI1" and "AI2". The first availability interval "AI1" is the overlap between the "S1.Q" quiescent phases of "Components J and K", while the second availability interval is the overlap between their "S2.Q" quiescent phases.

Figure 5.11 also shows that the requirement interval "RI1" associated with "Activity M" is covered by the second availability interval "AI2". This implies that the commencement of "Activity M" is contained by the second availability interval "AI2". Furthermore, since the quiescent phase "S4.Q" of "Component J" is later than "S4.Q" of "Component K", "S4" of "Component J" is the bottleneck state that constrains the commencement of "Activity M". This means that if "S4" of "Component J" can be brought forwards, the availability interval "AI2" can be accordingly start earlier for maintaining the stability of "Components A and B", leading to earlier commencement of "Activity M".

Identifying bottleneck state can aid planning engineers in reducing construction periods with respect to advancing bottleneck states that constrains the commencement of some critical activities. Moreover, the transition of a bottleneck state may be constrained by other component states through various types of state relationships. This means that planning engineers should proactively control the progress of upstream works in order to prevent delay of the bottleneck states.

## 5.5 Concluding Remarks

This chapter presents four analysis methodologies. The first and second methodologies can be used for evaluating the temporal and spatial interactions a single intermediate function, respectively. The third methodology extend the first methodology from evaluating the temporal interaction inside a single intermediate function to analyzing the temporal interactions between the user and providers associated with a set of compatible intermediate function requirements. The fourth methodology is used to identify bottleneck states that constrain the commencement of the associated construction activities.

Using the first three analyses, the temporal and spatial perspectives of intermediate function requirements can be evaluated for detecting unfulfilled requirements. Particularly, these analyses imply that the concurrent relationships, which are inadequately evaluated in traditional CPM schedules, can be analyzed with respect to the temporal relationships between/among component states. In this way, more scheduling conflicts can be detected, leading to improved constructability of a facility project.

The fourth methodology helps planning engineers locate the constraints of construction activities, especially those critical ones, with respect to bottleneck states. In this way, more constraints of construction activities can be identified from the intermediate function viewpoint. This information can also guide planning engineers to proactively control the completion of the upstream works that may delay the downstream bottleneck states. Moreover, some bottleneck states can be advanced in order to shorten the construction period, leading to better executability of construction schedules.

# CHAPTER 6 4D-iFAST PROTOTYPE

4D intermediate Function AnalysiS Tool (4D-iFAST) is a research prototype to implement the developed integration framework and the analysis methodologies as well as 4D simulation. The potential benefits from using 4D simulation to facilitate intermediate function analysis are first explored in this chapter. Subsequently, the main data structure of the prototype is represented by the class diagram. Based on the data structure, this chapter introduces the reasoning mechanism of the inference engine and the simulation mechanism of the 4D simulation engine. The existence vector concept and the Boolean operations on existence vectors are developed for speeding up the inference engine and the simulation engine, which are two kernel modules of the prototype. Lastly, some typical interface windows are demonstrated for illustrating the key functions of 4D-iFAST.

## 6.1 4D Simulation Environment for Intermediate Function Analysis

A 4D Model integrates both the three-dimensional geometric attribute and the temporal attribute for visualizing construction sequences. Several 4D simulation tools and prototypes have been studied in the past two developed. Previous studies indicated that these 4D simulation tools facilitate the constructability analysis mainly from three aspects, namely integration, visualization, and time-space analysis, as addressed in the literature reviews.

With the application of 4D simulation, the AEC industry can benefit from less rework, reduced trade interference, fewer change orders, and smoother work flows, leading to improved productivity and cost savings. It was reported that 4D simulation can help some companies save up to 45 percent of expenditure on change orders (Sheppard 2004). Another benefit of using 4D simulation is the better understanding

of the construction progress from all the project participants, which help them collaboratively schedule their tasks.

The previous research on 4D simulation implies that 4D simulation tools may also facilitate intermediate function analysis. The present study attempts to incorporate the 4D simulation function into the software prototype 4D-iFAST, which stands for **4D- I**ntermediate **F**unction **A**nalysi**s T**ool. One purpose of this prototype is to explore the feasibility and capability of using 4D simulation for facilitating the intermediate function analysis in terms of identifying the function user and function provider and exploring better engineering solutions for intermediate function requirements. This research may also help software vendors to enhance and customize their present products with respect to intermediate function analysis.

The potential contribution of 4D simulations to intermediate function analysis lies in three areas. Firstly, the 4D model integrates the spatial information stored in 3D CAD model and the temporal information stored in process model and in-progress product model as well as resource model. This means that the 4D model plays the role of a kernel to reference the spatial attributes and the temporal attributes distributed in the product, process, resource, and intermediate function models.

Secondly, the simulation frames can help designers and constructors to identify the intermediate function requirements, and facilitate their exploration for better engineering solutions to fulfill the intermediate function requirements. For example, through studying the 4D simulation scenarios, the glass work subcontractor may explore the feasibility of using the scaffold erected by the steel structure subcontractor, which, if feasible, could save the cost and time spent on additional temporary facilities and also accelerate the delivery of site space to the downstream cladding subcontractor.

Lastly, 4D simulation can help develop the spatio-temporal criterion matrix for verifying the spatial interactions between function users and function providers. Additionally, the developed 4D model can also be used to automatically analyzing temporal topological between the space entities. Specifically, the 4D model can be used to verify the accessibility of work faces with respect to the temporal "meet" relationship between a user space entity and a provider space entity. Meanwhile, based on the 4D model, the time-space conflicts can also be detected by locating the temporal intersection relationships between a pair of temporal space entities (Song and Chua, 2005).

## 6.2 Conceptual Architecture of 4D-iFAST Prototype

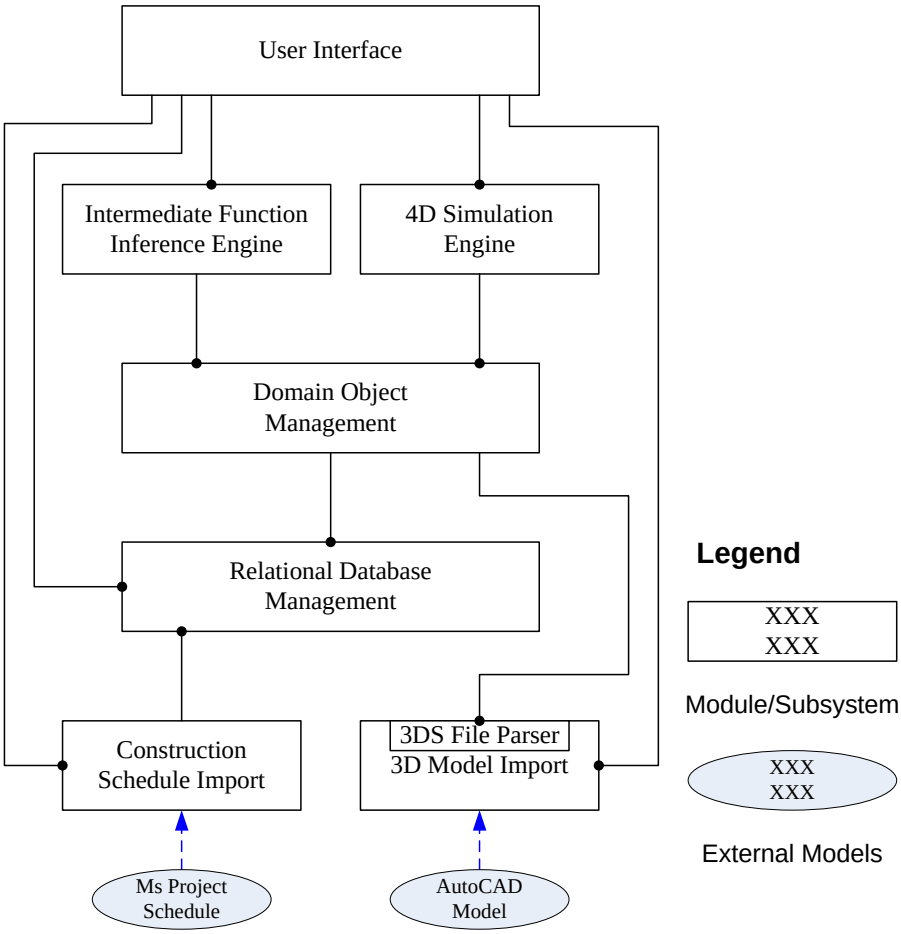



**Figure 6.1 Conceptual Architecture of 4D-iFAST**

4D-iFAST is developed mainly using the programming tool Delphi 5, and the domain database and intermediate function knowledge base are developed using Access 2003 Desktop Database. The rendering engine for 4D simulation is developed using OpenGL API. The construction schedule is developed using Ms Project 2003, and the 3D space model is created using AutoCAD 2000 and exported as a 3DS model.

Figure 6.1 illustrates the conceptual architecture of the developed 4D-iFAST prototype. This architecture highlights the key subsystems and the major dependency relationships between them. The architecture of the 4D-iFAST prototype is structured into five layers from top to bottom as shown in Figure 6.1. A subsystem in the upper layer may depend on the subsystems in the lower layers, but the subsystems in the lower layers do not depend on those in the higher layers. In this way, the dependency relationships between the subsystems can be decoupled to reduce the complexity of the prototype structure.

The bottom layer comprises two import applications. The “Construction Schedule Import” subsystem is developed for importing Ms Project schedule data into the Access database using the ActiveX Data Object (ADO) technology, while the “3D Model Import” subsystem is used to import the 3DS model, which is developed and exported by the AutoCAD application, into the “Domain Object management” subsystem for generating the space entity objects. The conversion from 3DS model to space entity objects is realized by a 3DS file parser module mainly written in Delphi 5.

The “Relational Database Management” subsystem manages all the data input from the user interface and the construction schedule data imported from MS Project application. Besides the construction schedule database, this subsystem comprises

another four relational databases: IPPCM, resource, intermediate function, and intermediate function knowledge databases.

The “Domain Object Management” subsystem is responsible for generating the corresponding domain objects using the record data stored in the databases, and also for retrieving and destroying the domain objects temporarily stored in the computer memory. Accordingly, an object buffer management module should also be developed to control the dynamic construction and destruction of the domain objects according to the inference and simulation requirements. These domain objects will be used by the “Intermediate Function Inference Engine” for evaluating intermediate functions and also are required by the “4D Simulation Engine” for visualizing the construction progress.

The “Intermediate Function Inference Engine” subsystem is the most important part of the prototype. It implements the analysis methodologies developed in Chapter 5 for reasoning out the unfulfilled intermediate function requirements from both temporal and spatial perspectives. The “4D Simulation Engine” is another important module for generating 4D simulation frames. This simulation engine implements the mechanism addressed in the following Section 6.6

The “User Interface” subsystem mainly comprises the interactive windows for importing, inputting, and editing the data for establishing the integrated information framework. On the other hand, this subsystem also comprises the interface windows for publishing, previewing, and printing the results of inferring intermediate functions and also for rendering the 4D simulation frames. Some of the interface windows will be shown and explained in the following section 6.7.

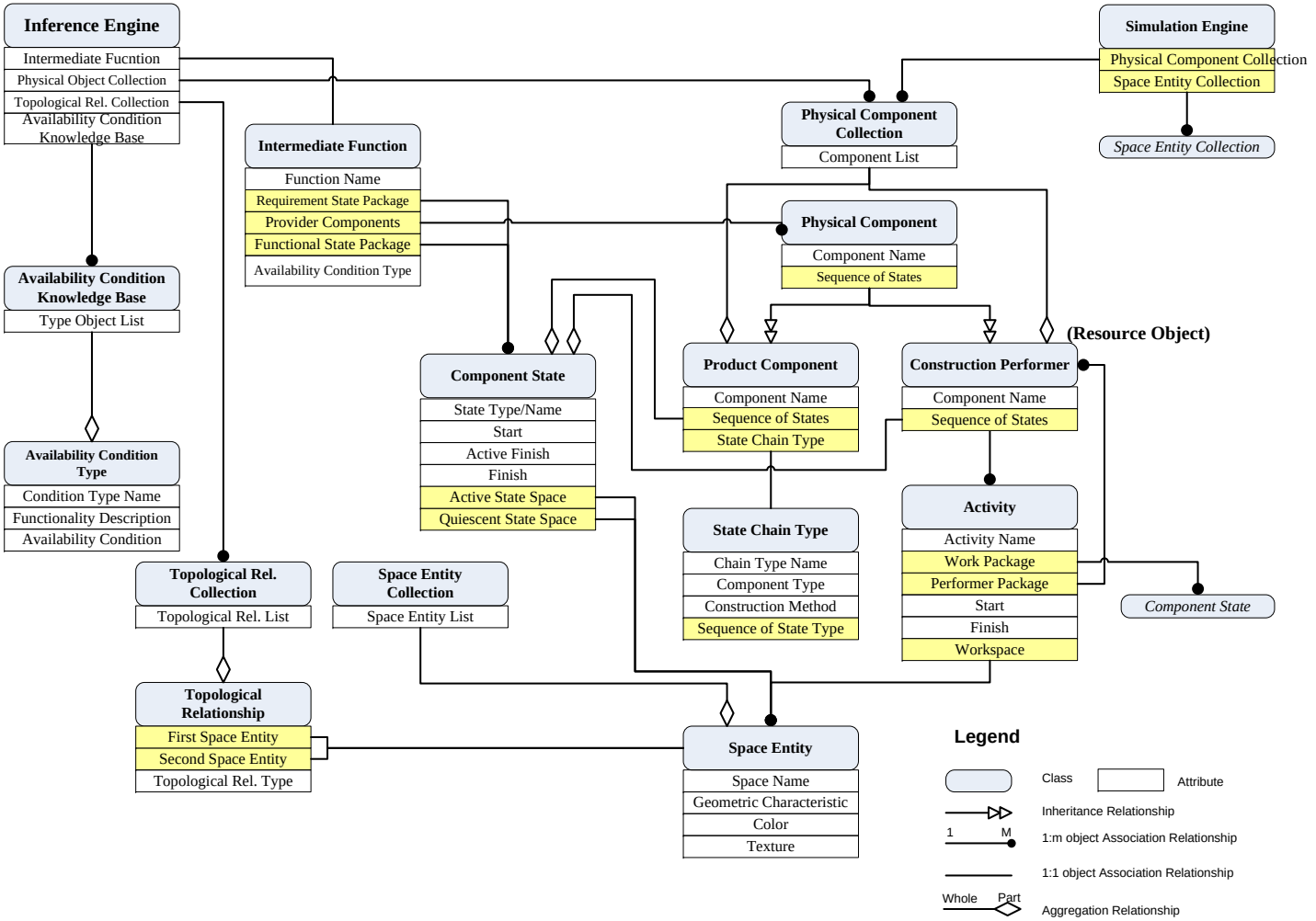



**Figure 6.2 Component-Relationship Structure of 4D-iFAST System**

## 6.3 Component-Relationship Structure of 4D-iFAST System

Figure 6.2 shows the component-relationship structure of the 4D-iFAST prototype using the class diagram. This Figure only shows the classes/types of the system objects related to the intermediate function inference and the 4D simulation due to space limitation. The class diagram illustrates the key modeling elements in five project perspective models (intermediate function, product, resource, process, and space) as well as the intermediate function knowledge and also represents the association and aggregation relationships among the classes. These classes and the relationships between them are necessary for understanding the inference mechanism and simulation mechanism addressed in the succeeding sections.

The *Activity* class is a fundamental modeling element in the process model. Its name attribute should be kept unique among a collection of activities. Each activity has both *start* and *finish* attributes that are produced by the external MS Project application. The *workspace* attribute can associate with one or more space entity(ies) for representing the workspace required by the associated performers. The *work package* and *performer package* attributes are used for establishing association relationships with product model and resource (construction performer) model, respectively. Specifically, the *work package* attribute can reference one or more *component state* objects for depicting the work content, while the *performer state package* can reference one or more *construction performer*s in the resource model.

The *physical component* is the super-class (parent class) of both *product component* and *construction performer* classes. It represents the common attributes (temporal and spatial attributes) and behaviors of *product components* and *construction performers*. Accordingly, the *physical component collection* class represents a collection of physical component. Figure 6.2 shows that the *physical*

*component collection* class can aggregate both product components and construction performers. These two types of components are often associated with an intermediate function to describe its user and provider. The super class *physical component* makes it possible to aggregate both product and resource component into one object list, which is represented by the *component list* attribute of the *physical component collection* class. In this way, the data structure design of the inference engine as well as the simulation engine can be simplified.

Figure 6.2 shows that the *physical component* class has two attributes: *component name* and *sequence of state*. The former is an identifier to distinguish the product/resource components in a component list, and therefore should be unique. The latter is a sequence of *component state* objects to represent either the construction life cycle of a *product component* or the execution history of a *construction performer*.

The *product compone*nt class is the subclass (child class) of the *physical component*. So the former inherits the two attributes of the latter, and adds another attribute *state chain type*, which references a *state chain type* object (see Figure 6.2). The most important attribute of the *state chain type* class is *sequence of state type*, which can be used for automatic generation of a default state chain as the value of the *sequence of states* attribute of a product component object. This can facilitate the input of the product component data. The *sequence of states* attribute represents the construction life cycle of a product component in terms of a chain of component states. Similar to the *product compone*nt class, the *construction performer* class also inherits the two attributes of its super-class *physical component*. The *sequence of states* herein represents the execution history of a construction performer in terms of a series of performer states, which are the activities that employ the construction performer.

The *component state* class acts as the bridge to link the product model with the process, space, and intermediate function models. Besides the state type/name attribute for identifying a component state object, this class also has three temporal attributes (*start*, *active finish*, and *finish*) and two spatial attributes (*active state space* and *quiescent state space*). The *start* and *active finish* of a component state determine the temporal interval of its active phase, while the *active finish* and *finish* determine its quiescent phase. In other words, the *active finish* is the point of time which divides the state duration the active and quiescent phases. Since a construction performer has zero-duration quiescent phase, its *active finish* time is equal to its *finish* time.

Each of the two state space attributes of the *component state* class may reference a number of space entities to represent the space occupied by an in-progress product component. The *active state space* attribute describes the space occupation during the *active state phase*. Similarly, the *quiescent state space* attribute describes the space occupation during the *quiescent state phase.* In this way, the difference in space utilization between the active and quiescent phases can be distinguished for better description of component states. For example, the space occupied by a portable platform can be represented by the boundary of the moving path during its "Moving.A" active phase, whereas it only occupies its physical volume during its "Moving.Q" quiescent phase, which is a much smaller boundary than the moving path. Additionally, the active state phase of a construction performer frequently references the *workspace* of the corresponding activity as its *active state space*.

The *space entity* class can depict either the physical volume of a fixed product component, the workspace of an activity, or the path space of moving object. The *space name* attribute is used as the identifier for retrieving a specific space entity in the s*pace entity collection*. The geometric characteristic attribute describes the shape

and location of the space entity in terms of *triangulated surfaces* that are imported from the 3DS model. In a 3DS model, all the surfaces of a space entity have been triangulated. The value of the *color* and *texture* attributes will be automatically changed by the *simulation engine* for visualizing in-progress configuration of the associated product components.

The *topological relationship* class represents the binary spatial relationships between two 3D space entities denoted by the *first space entity* and the *second space entity*. The topological relationship type attribute records the derived topological relationship type. There are 3 types of topological relationships, namely “Disjoint”, “Meet” and “Intersect”. All the derived topological relationships will be stored in the *topological relationship collection*, which is a list of topological relationship objects.

The *availability condition type* class is designed to implement the schema of availability condition knowledge defined in Chapter 4. In this way, the availability conditions of various types of intermediate functionalities can be represented by a set of *availability condition type* objects stored in the *availability condition knowledge base*. The *availability condition type* class has three attributes: *condition type name*, *functionality description*, and *availability condition*. The *condition type name* is the identifier for uniquely identifying a specific *availability condition type* object, and the *functionality description* attribute depicts the role or purpose of the intermediate functionality. The most important *availability condition* attribute stores a list of functional state types to abstract the in-progress engineering behaviors of the corresponding (product) component types.

The *intermediate function* class abstracts the intermediate function requirements. The *function name* attribute is the unique identifier of an intermediate function object. The construction states of the function user and the function provider

are depicted in terms of the *requirement state package* attribute and the *functional state package* attribute, respectively. Both packages reference a group of component state objects, which are aggregated in the corresponding *product components* or *construction performers*. The *availability condition type* attribute is used to reference the corresponding availability condition knowledge, which can be parsed by the *inference engine* for automatically generating *functional state package* using the component data stored in the *provider components* attribute. This attribute often references several *physical component* objects. This automatic assignment will be explained in Section 6.5.1.

The *inference engine* for evaluating the intermediate functions is also represented as a class. From this class, only one inference engine object will be constructed in the 4D-iFAST application. Its *intermediate function* attribute indicates the intermediate function that will be evaluated by the inference engine. The inference engine references two object lists through the *physical component collection* attribute and the *topological relationship* attribute in order to retrieve temporal and spatial information for evaluating the temporal and spatial interaction between the function user and the function provider. The inference engine also references the *availability condition knowledge base* for retrieving the *availability condition type* object.

The *simulation engine* class has two attributes: *physical component collection* and *space entity collection*. These two attributes can be used to derive the 4D model, which is essentially a system of temporal space entities. Each temporal space entity in the 4D model has an existence period attribute to depict when the space enters, exits as well as re-enters the temporal space system during the entire construction period. The derivation of the 4D model and its simulation mechanism will be explained in Section 6.6.

## 6.4 Existence Vector and Boolean Operations

### 6.4.1 Existence Vector Concept

A temporal interval can be represented by an existence vector instead of a pair of time points (start and finish points) as defined in Section 3.3.4. An *existence vector* is an *n*-element vector with each element denoting the existence of an object within a finite time interval. Each element in the existence vector is a Boolean value, where 1/true denotes the existence of the object in the system while 0/false denotes its absence. If all the finite intervals of an existence vector are 0/false, such an existence vector is called "zero" existence vector in the present study.

A *finite time interval* (FTI) is defined as a short time interval for segmenting a period of study, during which the attributes of an object can be assumed to be unchanged. In this way, a continuous study period can be segmented into an array of discrete finite time intervals. A finite time interval can be 1-week, 1-day, or even 1-hour depending on the precision and the nature of the problem studied. The size of an existence vector is therefore $T/t_f$, where T is the period of the study and $t_f$ the finite time interval. For example, assuming that the construction period T of a project is from Days 1 to 10 and the finite time interval $t_f$ is 1 day, the active phase of a component state "S1" which starts at the start of Day 3 and finishes at the end of Day 5 can be represented as an 10-element existence vector (0,0,1,1,1,0,0,0,0,0).

Moreover, compared with a temporal interval which can only represent a continuous period, an existence vector can be used to represent either a continuous temporal interval or a discontinuous time-window containing several intervals. This characteristic is very useful for evaluating the temporal interaction as well as other interval algebra computations for intermediate function analysis. Specifically, a state

duration or activity/quiescent duration is a temporal interval, while a requirement time-window, an availability time-window, a non-matching time-window, or the existence period of a temporal space entity can be a discontinuous time-window. The computation of these discontinuous time-windows can be simplified by using existence vectors and Boolean operations on them.

### 6.4.2 Boolean Operations between Two Existence Vectors

There are altogether three types of fundamental *Boolean operations* between two existence vector operands that are useful for the analysis of intermediate functions. They are Boolean union, Boolean intersection, and Boolean cut. The Boolean union operation ascertains all the non-zero elements from two existence vector operands as follows:

$$\begin{aligned}
&\mathbf{EV_a}(\mathbf{f}_1,\ldots,\mathbf{f_i},\ldots,\mathbf{f_n}) \cup \mathbf{EV_b}(\mathbf{g}_1,\ldots,\mathbf{g_i},\ldots,\mathbf{g_n}) \\
&= \mathbf{EV_r}(\mathbf{h}_1,\ldots,\mathbf{h_i},\ldots,\mathbf{h_n}) \\
&\mathbf{h_i} = \mathbf{f_i} \cup \mathbf{g_i} \quad \forall\ \mathbf{i} \in \mathbf{\{1, \ldots, n\}} \\
&\mathbf{h_i} = \begin{cases} 0 & \textbf{if } \mathbf{f_i} = 0 \textbf{ and } \mathbf{g_i} = 0 \\ 1 & \textbf{else} \end{cases}
\end{aligned} \qquad \textbf{(6.1)}$$

Where $EV_a(f_1, \ldots, f_i \ldots, f_n)$ and $EV_b(g_1, \ldots, g_i \ldots, g_n)$ are two arbitrary existence vectors whose sizes are n, $EV_r(h_1, \ldots, h_i \ldots, h_n)$ the resultant existence vector of the Boolean union operation, $f_i$ the $i^{th}$ element (at the $i^{th}$ finite time interval) of $EV_a$, $g_i$ the i$^{th}$ element (at the $i^{th}$ finite time interval) of $EV_b$, and $h_i$ the $i^{th}$ element (at the $i^{th}$ finite time interval) of $EV_r$. Eq. (6.1) indicates that at an arbitrary finite time interval *i*, the resultant existence vector has the value of 0, only if both existence vector operands have the values of 0. Otherwise, the Boolean union resultant will be assigned the value of 1 at the $i^{th}$ finite time interval.

The Boolean intersection operation ascertains the overlap of non-zero elements between two existence vector operands. The Boolean intersection formula is presented as:

$$\begin{aligned}
&\mathbf{EV_a}(\mathbf{f_1},\ldots,\mathbf{f_i},\ldots,\mathbf{f_n}) \cap \mathbf{EV_b}(\mathbf{g_1},\ldots,\mathbf{g_i},\ldots,\mathbf{g_n}) \\
&= \mathbf{EV_r}(\mathbf{h_1},\ldots,\mathbf{h_i},\ldots,\mathbf{h_n}) \\
&\mathbf{h_i} = \mathbf{f_i} \cap \mathbf{g_i}, \quad \forall\, \mathbf{i} \in \mathbf{\{1, \ldots, n\}} \\
&\mathbf{h_i} = \begin{cases} 1 & \textbf{if } \mathbf{f_i} = 1 \textbf{ and } \mathbf{g_i} = 1 \\ 0 & \textbf{else} \end{cases}
\end{aligned} \qquad \textbf{(6.2)}$$

Where $EV_a(f_1, \ldots, f_i \ldots, f_n)$ and $EV_b(g_1, \ldots, g_i \ldots, g_n)$ are two arbitrary existence vectors whose sizes are n, $EV_r(h_1, \ldots, h_i \ldots, h_n)$ the resultant existence vector of the Boolean intersection operation, $f_i$ the $i^{th}$ element (at the $i^{th}$ finite time interval) of $EV_a$, $g_i$ the $i^{th}$ element (at the $i^{th}$ finite time interval) of $EV_b$, and $h_i$ the $i^{th}$ element (at the $i^{th}$ finite time interval) of $EV_r$. Eq. (6.2) indicates that at an arbitrary finite time interval $i$, the resultant existence vector has the value of 1, only if both existence vector operands have the value of 1. Otherwise, the Boolean intersection resultant will be assigned the value of 0 at the $i^{th}$ finite interval unit.

The Boolean cut operation ascertains the finite time intervals where non-zero elements in the first existence vector operand corresponding to the zero elements in the second operand. The Boolean cut formula is presented as follows:

$$\begin{aligned}
&\mathbf{EV_a}(\mathbf{f_1},\ldots,\mathbf{f_i},\ldots,\mathbf{f_n}) - \mathbf{EV_b}(\mathbf{g_1},\ldots,\mathbf{g_i},\ldots,\mathbf{g_n}) \\
&= \mathbf{EV_r}(\mathbf{h_1},\ldots,\mathbf{h_i},\ldots,\mathbf{h_n}) \\
&\mathbf{h_i} = \mathbf{f_i} - \mathbf{g_i} = \mathbf{f_i} \cap \mathbf{not}(\mathbf{g_i}) \quad \forall\, \mathbf{i} \in \mathbf{\{1, \ldots, n\}} \\
&\mathbf{h_i} = \begin{cases} 0 & \textbf{if } \mathbf{f_i} = 1 \textbf{ and } \mathbf{g_i} = 1 \\ 1 & \textbf{if } \mathbf{f_i} = 1 \textbf{ and } \mathbf{g_i} = 0 \\ 0 & \textbf{if } \mathbf{fi} = 0 \textbf{ and } \mathbf{g_i} = 1 \\ 0 & \textbf{if } \mathbf{fi} = 0 \textbf{ and } \mathbf{g_i} = 0 \end{cases}
\end{aligned} \qquad \textbf{(6.3)}$$

Where $EV_a(f_1, \ldots, f_i \ldots, f_n)$ and $EV_b(g_1, \ldots, g_i \ldots, g_n)$ are two arbitrary existence vectors whose sizes are n, $EV_r(h_1, \ldots, h_i \ldots, h_n)$ the resultant existence vector of the

Boolean cut operation, $f_i$ the $i^{th}$ element (at the $i^{th}$ finite time interval) of $EV_a$, $g_i$ the i$^{th}$ element (at the $i^{th}$ finite time interval) of $EV_b$, and $h_i$ the $i^{th}$ element (at the $i^{th}$ finite time interval) of $EV_r$. Eq. (6.3) indicates that at an arbitrary finite time interval *i*, the resultant existence vector has the value of 1, only if the first existence vector has the value of 1 and the second existence vector has the value of 0. Otherwise, the Boolean cut resultant will be assigned the value of 0 at the $i^{th}$ finite time interval.

These three fundamental Boolean operations can be used to compute various types of time-windows during intermediate function analysis. Specifically, the Boolean union operation between two existence vectors can be used to compute the requirement time-window following Eq. (5.1). Likewise, the Boolean intersection operation as well as the Boolean union operation can be used for deriving the availability time-window following Eq. (5.2). The Boolean cut operation can be used to derive non-matching time-windows following Eq. (5.3). Additionally, the existence periods of both temporal space entities and temporal topological relationship can also be derived by using Boolean union and intersection operations following the Eq. (5.4) and Eq. (5.5), respectively.

### 6.4.3 Boolean Operations on a Set of Existence Vectors

Two additional Boolean operations, namely "multiple-union" (denoted by $\cup_{\mathbf{m}}$) and "multiple-intersection" (denoted by $\cap_{\mathbf{m}}$), are defined to manipulate a set of existence vectors. The formula of the multiple-union is presented as follows:

$$\mathbf{EV_r(f_1,...,f_k,...f_n) = \bigcup_{i=1}^{m} m\left(EV_i(f_1,...,f_k,...f_n)\right)}$$

$$\mathbf{EV_r(f_k) = \bigcup_{i=1}^{m} EV_i(f_k),\ \forall\ i \in \{1, \ldots, m\},\ and\ k \in \{1, \ldots, n\}} \qquad \mathbf{(6.4)}$$

Where $EV_i(f_1, \ldots, f_k, \ldots, f_m)$ is the $i^{th}$ existence vector operand whose size is m, $EV_i(f_k)$ the $k^{th}$ element of the $i^{th}$ existence vector operand, $EV_r(f_k)$ the $k^{th}$ element of the multi-union resultant existence vector. Eq. (6.4) indicates that the $k^{th}$ element (at the $k^{th}$ finite time interval) of the multiple-union resultant $EV_r$, i.e. $EV_r(f_k)$, is the Boolean union resultant of all the $k^{th}$ elements of *m* existence vector operands. Therefore, at the $k^{th}$ finite time interval, only when all the $k^{th}$ elements of *m* existence vector operands have the values of 0, will the multiple-union resultant have the value of 0. Otherwise, the multiple-union resultant has the value of 1 at the $k^{th}$ finite time interval.

Similarly, the formula of the multiple-intersection is presented as follows:

$$\mathbf{EV_r(f_1,...,f_k,...f_n) = \bigcap_{i=1}^{m} \left(EV_i(f_1,...,f_k,...f_n)\right)}$$

$$\mathbf{EV_r(f_k) = \bigcap_{i=1}^{m} EV_i(f_k),\ \forall\ k \in \{1, \ldots, n\} \text{ and } i \in \{1, \ldots, m\}} \qquad \mathbf{(6.5)}$$

Where $EV_i(f_1, \ldots, f_k, \ldots, f_m)$ is the $i^{th}$ existence vector whose size is m, $EV_i(f_k)$ the $k^{th}$ element of the $i^{th}$ existence vector operand, $EV_r(f_k)$ the $k^{th}$ element of the multi-intersection resultant existence vector. Eq. (6.5) indicates that the $k^{th}$ element (at the $k^{th}$ finite time interval) of the multiple intersection resultant, i.e. $EV_r(f_k)$, is the Boolean intersection result of all the $k^{th}$ elements of *m* existence vector operands. It can be derived that at the $k^{th}$ finite time interval, if any existence vector operand has the value of 0, the $k^{th}$ element of the multi-intersection resultant is assigned the value of 0. Only when all the $k^{th}$ elements of *m* existence vector operands have the values of 1, will the $k^{th}$ element of the multi-intersection resultant have the value of 1.

The implementation of the inference engine can be enhanced by using existence vectors and Boolean operations between them (Song and Chua, 2005). Using the two additional Boolean operations $\cup_{\mathbf{m}}$ and $\cap_{\mathbf{m}}$, the requirement and availability time-windows as well as the existence period of a temporal space entity

can be simplified when there are more than two associated states in the temporal computation. Additionally, the inference speed of temporal analysis for an intermediate function can be improved by using the Boolean operations since the computation of Boolean values are much faster than the computation of comparing integers or floats required by the traditional interval algebra. Moreover, the existence vector representation format together with the Boolean operations also makes it possible to implement spatio-temporal analysis without having to conduct 4D simulation (Song and Chua, 2005).

## 6.5 Inference Mechanism for Evaluating Intermediate Function

Figure 6.3 illustrates the inference mechanism for evaluating the temporal interaction of an intermediate function, comprising mainly four steps. Each step comprises several functions that can be concurrently implemented and synchronized in multiple parallel threads marked by circled labels. The first digit in each circled label defines the inference step where the thread is implemented. For example, the circle “1a” indicates that the “Retrieve Provider Components” function is implemented in the first inference step (see Figure 6.3). Additionally, The Figure shows that an engineer will enter the data for such attributes of an intermediate function as availability condition type, provider component, and requirement state package (marked by rectangles in Figure 6.3) in addition to the spatio-temporal criterion matrix. The development of the spatio-temporal criterion matrix can be facilitated by the 4D simulation.

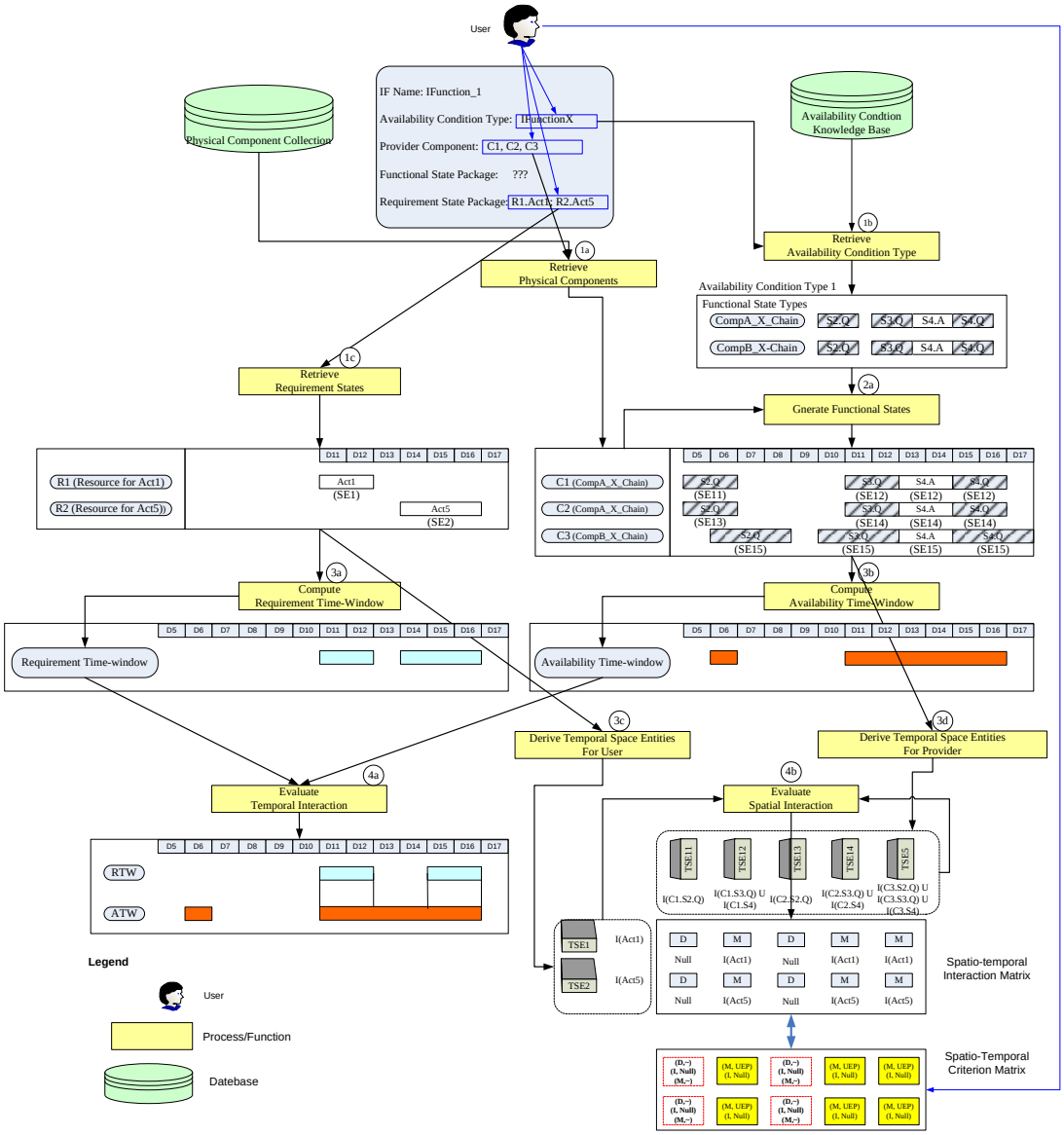



**Figure 6.3 Inference Mechanism for Evaluating Temporal Interaction**

The first inference step comprises three functions that will be concurrently implemented in three parallel threads. The “Retrieve Provider Components” function (implemented in the thread “1a”) is to retrieve the provider components from the physical component collection (see Figure 6.2) according to the names of the provider components entered by the user. Meanwhile, the “Retrieve Availability Condition Type” function (implemented in the thread “1b”) is to retrieve the availability condition type object from the availability condition knowledge base which stores the candidate availability conditions for different types of intermediate functions. The

retrieved provider components (product components or construction performers) and the retrieved availability condition type object are temporarily stored in the memory for use in the next inference step. The “Retrieved Requirement States” function (implemented in the thread “1c”) is to firstly retrieve the user components from the physical component collection according to the names of the component users (like “R1” in “R1.Act1” in Figure 6.3) and then retrieve the requirement states from the *sequence of states* attributes of the retrieved user components according to the state names (like “Act1” in “R1.Act1” in Figure 6.3).

The second inference step comprises only one function “Generate Functional States” (implemented in the thread “2a”). This function cannot be implemented in the first inference step since it requires the physical objects and the availability condition type object retrieved in the first step to automatically generate the functional state package, which is a list of component state names. In this way, the friendliness of the user interface can be improved since the users can save a lot of effort on manually entering the list of functional states. For example, the “Generate Functional States” function can automatically generate three functional states: “S2.Q”, “S3.Q”, and “S4” for the provider component “C1” according to the “Availability Condition Type 1” where a product component of the “CompA_X_Chain” type has these three functional states. The functional states of provider components “C2” and “C3” can be likewise generated. Similar to the function “Retrieve Requirement States”, the function “Retrieve Functional States” uses the generated list of component state names to locate and associate the corresponding component state objects from the physical component collection.

The third inference step comprises four parallel functions which are respectively implemented in four parallel threads marked by circled “3a”, “3b”, “3c”,

and “3d”. Using Eq. (6.4), the function “Compute Requirement Time-Window” (implemented by the thread “3a”) can compute the requirement time-window from the existence vectors of the requirement states retrieved in the second step. Similarly, using the Eqs. (6.4) and (6.5), the “Compute Availability Time-Window” function (implemented by the thread “3b”) can compute the availability time-window from the duration existence vectors of the functional states retrieved in the second step.

Meanwhile, the function “Derive Temporal Space Entities for User” (implemented in the thread “3c”) is responsible for deriving the temporal space entities associated with each component state/phase listed in the requirement state package. The time-independent space entities associated with each requirement state are first retrieved, and then its existence period can be derived from the corresponding requirement states using the Eqs. (6.4). Likewise, the function “Derive Temporal Space Entities for provider” (implemented in the thread “3d”) can derive the temporal space entities associated with each functional state from the associated functional states.

In the fourth and final inference step, two functions “Evaluate Temporal Interaction” and “Evaluate Spatial Interaction” are concurrently implemented in two parallel threads “4a” and “4b” for evaluating the fulfillment of the intermediate function requirement from the temporal and spatial perspectives, respectively. The function “Evaluate Temporal Interaction” is responsible for matching the requirement and availability time-windows respectively derived in the third inference step. The matching process uses the equation Eq. (6.3) to derive the non-matching time-window. If the resultant non-matching time-window is a “zero” existence vector the corresponding temporal interaction is available. Otherwise, the non-zero elements in

the non-matching time-window indicate the temporal interval(s) during which the temporal interaction is unavailable.

Simultaneously, the function "Evaluate Spatial Interaction" will be implemented in two sequential procedures. This function firstly produces the spatio-temporal interaction matrix from the derived user and provider temporal space entities using the approach introduced in Section 5.2.2. Then, it automatically compares the derived spatio-temporal interaction matrix against the spatio-temporal criterion matrix entered by planning engineers to identify undesirable temporal topological relationships. In this way, the unavailable spatial interaction between the function uses and the corresponding providers can be detected.

**Table 6.1 Temporal Data of Requirement/Functional States of "IFunction_1"**

| **Component States** | **Temporal Interval**<br>I(State/Phase Duration ) | **Existence Vector**<br>EV(I(State/Phase Duration )) |
|---|---|---|
| R1.Act1 (RS*) | I(11, 12) | (00000 00000 11000 00) |
| R2.Act5 (RS*) | I(14, 16) | (00000 00000 00011 10) |
| C1.S2.Q (FS*) | I(5, 6) | (00001 10000 00000 00) |
| C1.S3.Q (FS*) | I(11, 12) | (00000 00000 11000 00) |
| C1.S4 (FS*) | I(13, 16) | (00000 00000 00111 10) |
| C2.S2.Q (FS*) | I(5, 6) | (00001 10000 00000 00) |
| C2.S3.Q (FS*) | I(11, 12) | (00000 00000 11000 00) |
| C2.S4 (FS*) | I(13, 16) | (00000 00000 00111 10) |
| C3.S2.Q (FS*) | I(6, 8) | (00000 11100 00000 00) |
| C3.S3.Q (FS*) | I(10, 12) | (00000 00001 11000 00) |
| C3.S4 (FS*) | I(13, 17) | (00000 00000 00111 11) |

***Note: *RS* denotes a requirement state, while *FS* denotes a functional state.**

The intermediate function "IFunction_1" illustrated in Figure 6.3 is used as an example to demonstrate the interval algebra computation required in the aforementioned inference functions in terms of Boolean operations on existence vectors. It is assumed that the construction period is 17 work days, and the finite time

interval is 1 work day. Table 6.1 Lists the associated requirement and functional state/phases, their duration intervals and the corresponding existence vectors.

The user of “IFunction_1” comprises two construction performers “R1” and “R2” who requires the “IFunction_1” functionality when R1 executes the activity “Act1” and “R2” executes the activity “Act5”, while the provider of “IFunction_1” comprises three product components “C1”, “C2”, and “C3” along with their functional states/phases “S2.Q”, “S3.Q” and “S4”. The requirement time-window RTW(IFunction_1) can be produced using the equation Eq. (5.1) and the multiple-union (Eq. (6.4)) as follows:

$$\begin{aligned}
&\mathbf{EV(RTW(IFunction_1))}\\
&\mathbf{= EV(I(R1.Act1)) \cup EV(I(R2.Act5))}\\
&\mathbf{= (00000\ 00000\ 11000\ 00) \cup (00000\ 00000\ 00011\ 10)}\\
&\mathbf{= (00000\ 00000\ 11011\ 00)}
\end{aligned}$$

The resultant requirement time-window, which is an existence vector containing four true elements, denotes that the intermediate functionality is required at the eleventh, twelfth, fourteenth, and fifteenth finite intervals.

Similarly, the availability time-window ATW(IFunction_1) can be computed using the equation Eq. (5.2), following the multiple-union (Eq. (6.4)) and multi-intersection (Eq. (6.5)) operations, as follows:

$$\begin{aligned}
&\mathbf{EV(ATW(IFunction_1))}\\
&\mathbf{= \cap m\,(\cup m\,(EV(I(C1.S2.Q)), EV(I(C1.S3.Q)), EV(I(C1.S4))),}\\
&\qquad\mathbf{\cup m\,(EV(I(C2.S2.Q)), EV(I(C2.S3.Q)), EV(I(C2.S4))),}\\
&\qquad\mathbf{\cup m\,(EV((I(C3.S2.Q)), EV(I(C3.S3.Q)), EV(I(C3.S4))))}
\end{aligned}$$

**= $\cap_m$ ($\cup_m$ ((00001 10000 00000 00), (00000 00000 11000 00), (00000 00000 00111 10)),**

**$\cup_m$ ((00001 10000 00000 00), (00000 00000 11000 00), (00000 00000 00111 10)),**

**$\cup_m$ ((00000 11100 00000 00), (00000 00001 11000 00), (00000 00000 00111 11)))**

**= $\cap_m$ ((00001 10000 11111 10), (00001 10000 11111 10), (00000 11101 11111 11))**

**= (00000 10000 11111 10)**

The resultant availability time-window denotes that the intermediate functionality is available at the sixth and eleventh to fifteenth finite intervals.

Then, following Eq. (5.3), the non-matching time-window can be computed by Boolean cutting the EV(ATW) from the EV(RTW), which is shown in the following:

**EV(NMTW(IFunction_1))**

**= EV(RTW(IFunction_1)) - EV(ATW(IFunction_1))**

**= (00000 00000 11011 00) - (00000 10000 11111 10)**

**= (00000 00000 00000 00)**

The aforementioned non-matching resultant is a zero existence vector, indicating that the temporal interaction between the function user and the function provider is available.

On the other hand, the derivation of the existence periods of both temporal space entities and temporal topological relationships can also be facilitated by using the Boolean operations on existence vectors. For example, the existence period of the temporal space entity “TSE14”, referenced by both “S3.Q” and “S4” states of the provider component “C2” in Figure 6.3, can be computed as follows:

**EV(EP(TSE14))**

**= EV(I(C2.S3.Q)) ∪ EV(I(C2.S4))**

**= (00000 00000 11000 00) ∪ (00000 00000 00111 10)**

**= (00000 00000 11111 10)**

The existence period of the temporal topological relationship between “TSE1” and “TSE14” can be computed via the Boolean intersection of their existence periods in terms of existence vectors. The existence vector of the temporal space entity “TSE1” is equal to the existence vector of the requirement state “R1.Act1” since “TSE1” is only reference by “R1.Act1”. Thus, the computation of the existence period of TTR(TSE1, TSE14) can be represented as follows:

**EV(EP(TTR(TSE1, TSE14)))**

**= EV(EP(TSE1)) ∩ EV(EP(TSE14))**

**= (00000 00000 11000 00) ∩ (00000 00000 11111 10)**

**= (00000 00000 11000 00)**

The resultant existence period indicates that the temporal topological relationship only exists in the eleventh and twelfth finite intervals since the user component “R1” and the provider component “C2” co-exist on Days 11 and 12.

## 6.6 4D Simulation Engine

The 4D simulation addressed in the beginning of this chapter is implemented by the 4D simulation engine, mainly comprising two functions: a *4D model generator* and a *frame filter*. The former is a function for automatically generating the system of temporal space entities, while the latter is a retrieving function to filter out the temporal space entities that should be rendered at the specified time frames.

A 4D model comprise a set of temporal space entities that are associated by both the IPPCM and the process model. The temporal work, path, and physical space entities in a 3D space model are associated with the product components and the construction performers aggregated in the physical component collection. The *4D model generato*r function retrieves these associate relationships for computing the

existence vector for each temporal space entity. The existence period of a temporal work/path space entity is the Boolean union of the duration existence vectors of the construction activities that reference the specified work/path space entity to describe their work/transportation boundaries. On the other hand, the existence period of a temporal physical space entity is the Boolean union of the duration existence vectors of the component states/phases that reference the specified physical space entity as their state spaces. In this way, a time-independent 3D space system can be converted into a 4D model by deriving their existence periods. Bases on such a 4D model, the simulation frames can be generated by the *frame filter* function.

Figure 6.4 shows an example to derive the existence vectors of four space entities “TSE1”, “TSE2”, “TSE3”, and “TSE4”, which are respectively referenced by the state spaces of two temporary components “T01” and “T02”. “T01” comprises two parts whose physical shapes are represented by “TSE1” and “TSE2”, respectively. The construction life cycle of “T01” indicates that the “TSE1” part is erected during the “S1” state, and then the “TSE2” part is erected on the top of “TSE1” during the “S2” state. Subsequently, the “TSE2” part will be dismantled during the third state “S3”, and then the “TSE1” part will be dismantled during the last state “S4”. Therefore, “T01.S4” does not reference any space entity. Accordingly, the existence vector of “TSE1” can be derived by the Boolean union of the existence vectors of the “T01.S1”, “T01.S2”, and “T01.S3” states, while the existence vector of “TSE2” is equal to the existence vector of the “T01.S2” state.

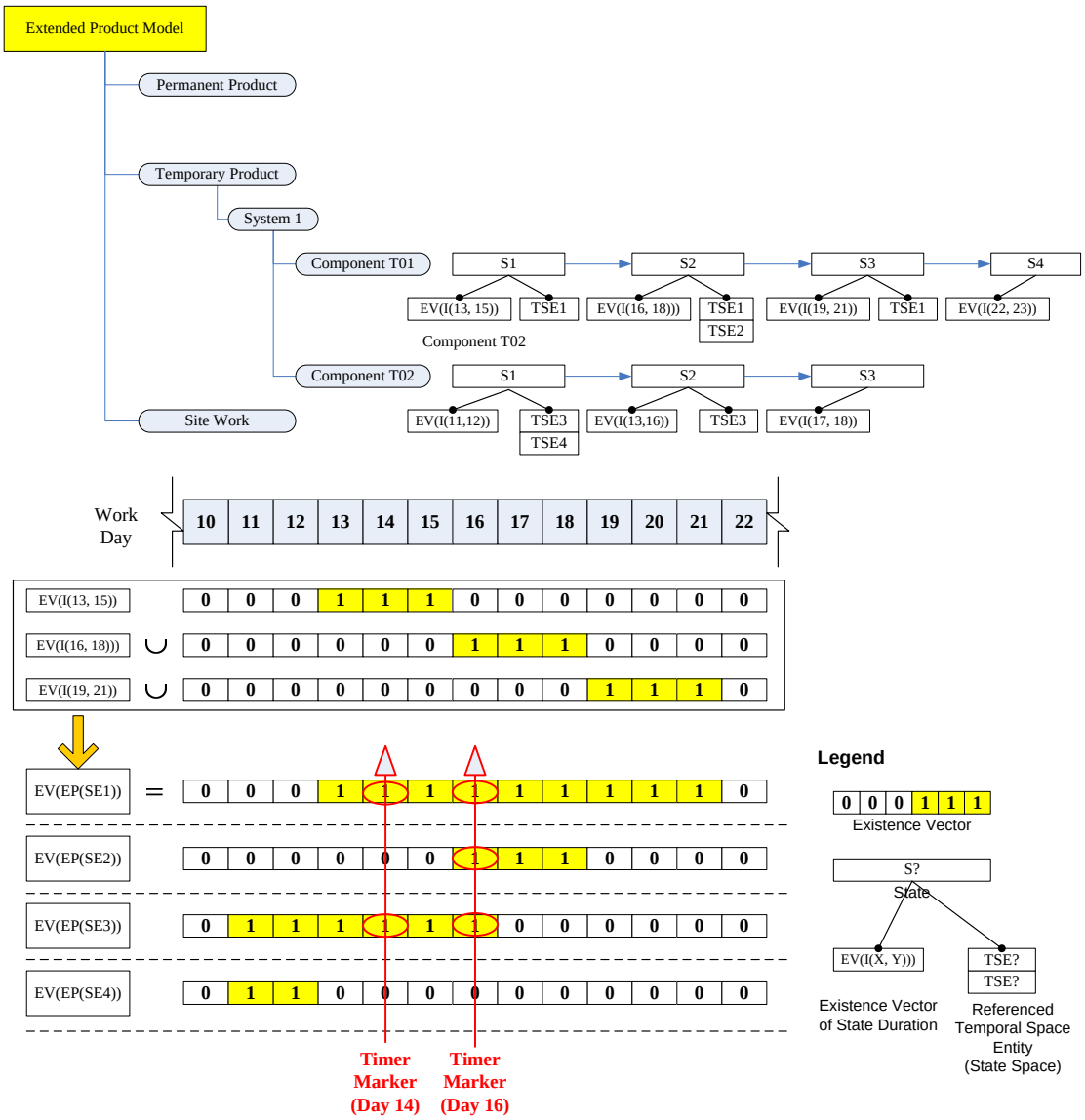



**Figure 6.4 Mechanism of 4D Simulation Engine**

On the other hand, the construction life cycle of “T02” comprises three states: “S1”, “S2”, and “S3”. The state space of state “T02.S1” references two physical space entities “TSE3” and “TSE4”, while the state space of state “T02.S2” only references one space entity “TSE3”, indicating that “TSE4” is demolished during the “T02.S2” state. During the third and last state “T02.S3”, the “TSE3” part will be demolished, so the state space of “T02.S3” references no space entity. Thus, the existence vector of “TSE4” is equal to that of the “T02.S1”, which is the only state that references

"TSE4", and the existence vector of "TSE3" is the Boolean union of the existence vectors of the associated states "T02.S1" and "T02.S2" (see Figure 6.4).

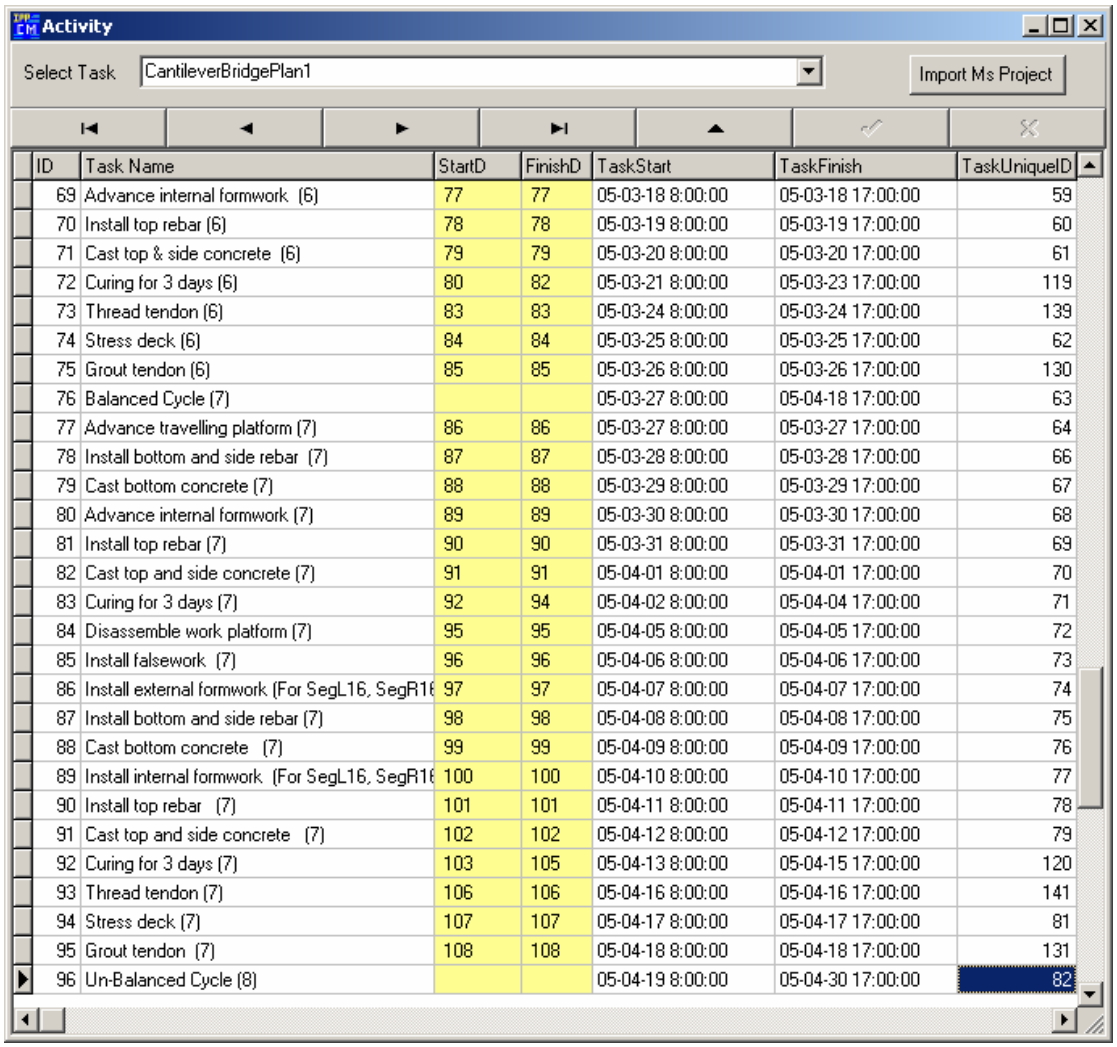
The "frame filter" is a simple function to retrieve the temporal space entities that should be rendered in a specified frame according to their existence vectors. For each temporal space entity in the 4D model that has been generated and stored in the computer memory, if its element at the time marker has the value of true/1, the corresponding temporal space entity is filtered into the rendering set. Otherwise, the temporal state entities will not be selected for rendering. Figure 6.4, for example, shows that when the time marker points "Day 14", i.e. the $14^{th}$ finite time interval of the period of study, only the temporal space entities "TSE1" and "TSE3" are filtered and rendered, but "TSE2" and "TSE4" are not. This is because "TSE1" and "TSE3" have the "true/1" elements at the $14^{th}$ finite time interval, but "TSE2" and "TSE4" have "0/false" value at the $14^{th}$ finite time intervals in their existence vectors, indicating that "TSE2" has not been erected and "TSE4" has been dismantled on Day 14 (see Figure 6.4). The Figure also shows when the time marker points Day 16, besides "TSE1" and "TSE3", "TSE2" is also filtered for rendering since the "TSE2" part of "T01" is erected on Day 16.

## 6.7 Typical User Interfaces

This section illustrates the user interfaces developed in 4D-iFAST, and simultaneously explains the major input and output at each interface. Generally, the intermediate function inference engine and the 4D simulation engine can get the required data and publish the inference and simulation results through these interfaces. Due to the space limitation of the present thesis, the illustration and explanation of some comparatively infrequent interfaces, like the database maintenance utilities, are

omitted. The introduction of the interfaces follows the workflow of using 4D-iFAST prototype system to conduct intermediate function analysis. The project case illustrated in most of the interfaces is a post-tensioned bridge structure constructed using the balance-cantilever construction method, which will be elaborated in the succeeding chapter.

### 6.7.1 In-Progress Product Core Model Interface




| ID | Task Name | StartD | FinishD | TaskStart | TaskFinish | TaskUniqueID |
|---|---|---|---|---|---|---|
| 69 | Advance internal formwork (6) | 77 | 77 | 05-03-18 8:00:00 | 05-03-18 17:00:00 | 59 |
| 70 | Install top rebar (6) | 78 | 78 | 05-03-19 8:00:00 | 05-03-19 17:00:00 | 60 |
| 71 | Cast top & side concrete (6) | 79 | 79 | 05-03-20 8:00:00 | 05-03-20 17:00:00 | 61 |
| 72 | Curing for 3 days (6) | 80 | 82 | 05-03-21 8:00:00 | 05-03-23 17:00:00 | 119 |
| 73 | Thread tendon (6) | 83 | 83 | 05-03-24 8:00:00 | 05-03-24 17:00:00 | 139 |
| 74 | Stress deck (6) | 84 | 84 | 05-03-25 8:00:00 | 05-03-25 17:00:00 | 62 |
| 75 | Grout tendon (6) | 85 | 85 | 05-03-26 8:00:00 | 05-03-26 17:00:00 | 130 |
| 76 | Balanced Cycle (7) | | | 05-03-27 8:00:00 | 05-04-18 17:00:00 | 63 |
| 77 | Advance travelling platform (7) | 86 | 86 | 05-03-27 8:00:00 | 05-03-27 17:00:00 | 64 |
| 78 | Install bottom and side rebar (7) | 87 | 87 | 05-03-28 8:00:00 | 05-03-28 17:00:00 | 66 |
| 79 | Cast bottom concrete (7) | 88 | 88 | 05-03-29 8:00:00 | 05-03-29 17:00:00 | 67 |
| 80 | Advance internal formwork (7) | 89 | 89 | 05-03-30 8:00:00 | 05-03-30 17:00:00 | 68 |
| 81 | Install top rebar (7) | 90 | 90 | 05-03-31 8:00:00 | 05-03-31 17:00:00 | 69 |
| 82 | Cast top and side concrete (7) | 91 | 91 | 05-04-01 8:00:00 | 05-04-01 17:00:00 | 70 |
| 83 | Curing for 3 days (7) | 92 | 94 | 05-04-02 8:00:00 | 05-04-04 17:00:00 | 71 |
| 84 | Disassemble work platform (7) | 95 | 95 | 05-04-05 8:00:00 | 05-04-05 17:00:00 | 72 |
| 85 | Install falsework (7) | 96 | 96 | 05-04-06 8:00:00 | 05-04-06 17:00:00 | 73 |
| 86 | Install external formwork (For SegL16, SegR1 | 97 | 97 | 05-04-07 8:00:00 | 05-04-07 17:00:00 | 74 |
| 87 | Install bottom and side rebar (7) | 98 | 98 | 05-04-08 8:00:00 | 05-04-08 17:00:00 | 75 |
| 88 | Cast bottom concrete (7) | 99 | 99 | 05-04-09 8:00:00 | 05-04-09 17:00:00 | 76 |
| 89 | Install internal formwork (For SegL16, SegR1 | 100 | 100 | 05-04-10 8:00:00 | 05-04-10 17:00:00 | 77 |
| 90 | Install top rebar (7) | 101 | 101 | 05-04-11 8:00:00 | 05-04-11 17:00:00 | 78 |
| 91 | Cast top and side concrete (7) | 102 | 102 | 05-04-12 8:00:00 | 05-04-12 17:00:00 | 79 |
| 92 | Curing for 3 days (7) | 103 | 105 | 05-04-13 8:00:00 | 05-04-15 17:00:00 | 120 |
| 93 | Thread tendon (7) | 106 | 106 | 05-04-16 8:00:00 | 05-04-16 17:00:00 | 141 |
| 94 | Stress deck (7) | 107 | 107 | 05-04-17 8:00:00 | 05-04-17 17:00:00 | 81 |
| 95 | Grout tendon (7) | 108 | 108 | 05-04-18 8:00:00 | 05-04-18 17:00:00 | 131 |
| 96 | Un-Balanced Cycle (8) | | | 05-04-19 8:00:00 | 05-04-30 17:00:00 | 82 |

**Figure 6.5 Import Process Schedule Data from Ms Project**

The first step to use 4D-iFAST is to import the process-oriented schedule developed using the MS Project 2000 or 2003 application. Figure 6.5 shows that such data fields as “Task ID”, “Task Name”, “Task Start”, “Task Finish”, and “Task

Unique ID" in the MS Project task table are imported into the process database developed in the 4D-iFAST. The "Task Unique ID" is the primary key of both the original task schedule table in MS Project and the imported task schedule table in 4D-iFAST. These two primary keys are crucial for synchronizing the changes in both task schedule tables. Additionally, the data table storing the imported schedule data is indexed by "Task ID", which represents the row number where the task lies in the Gantt chart grid as defined in the MS Project application.

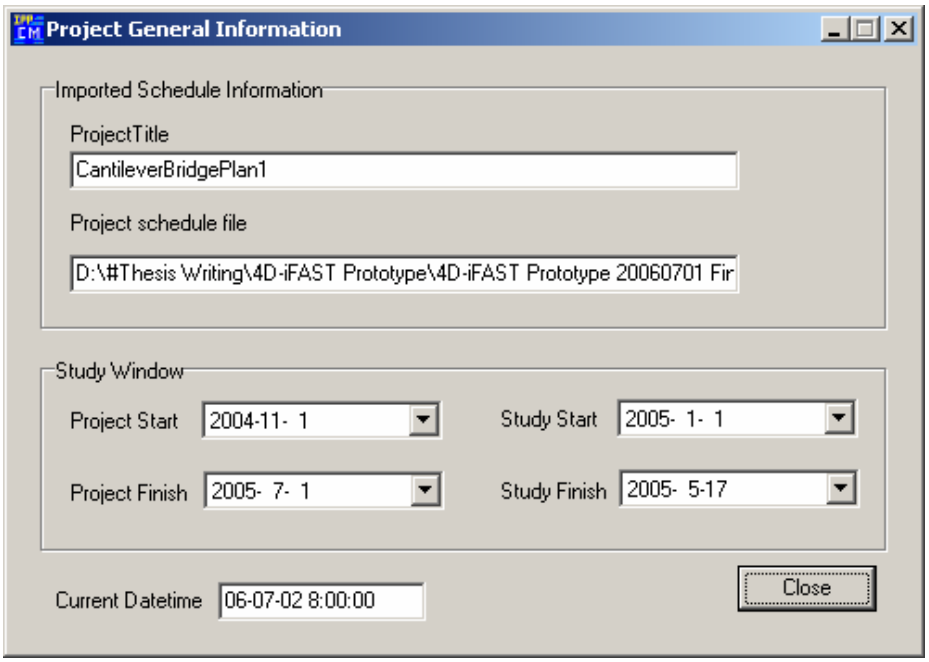



**Figure 6.6 Study Period and Construction Period**

Moreover, the "StartD" and "FinishD" attributes are the start and finish time points of each construction process in terms of the number of days from the "Study Start", which is start time of the study period for the intermediate function analysis. "StartD" and "FinishD" are automatically computed by the "Construction Schedule Importation" module (Figure 6.1). For example, Figure 6.6 shows that the construction period started on November 1$^{st}$, 2004 and finished on July 1$^{st}$, 2005, whereas the study period started on January 1$^{st}$, 2005 and finished on May 29$^{th}$, 2005. Figure 6.5 shows that "Install Bottom and Side Rebar (7)" ("ID 78") started on March,

28th, 2005, and its “StartD” is Day 87. Furthermore, assuming that the finite interval is one day, these “StartD” and “FinishD” attributes of each activity respective defines the start and finish finite intervals in the corresponding duration existence vector.

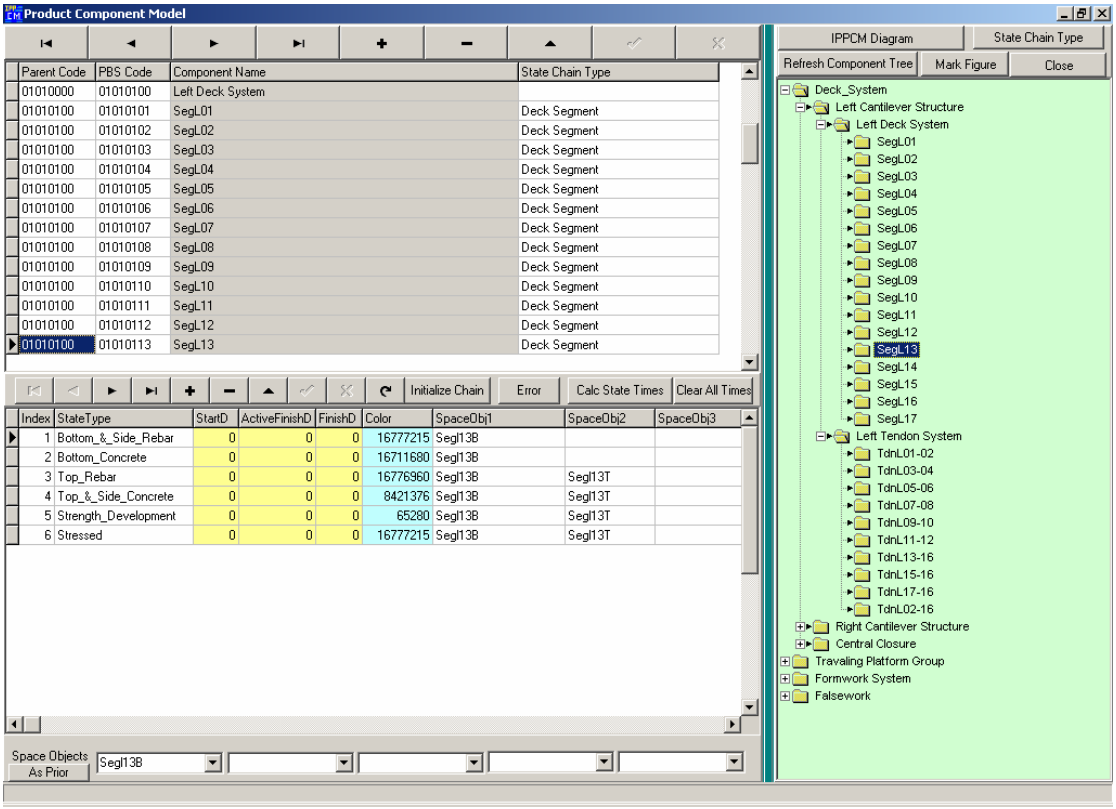



**Figure 6.7 Input of Product Hierarchy and Component States**

After the MS Project schedule has been imported, the user can continue to input the product decomposition hierarchy and then define the state chain for each product component in the hierarchy. The top left of Figure 6.7 illustrates the grid for developing the product decomposition hierarchy. The tree diagram on the right is automatically updated by the application as the product component hierarchy is updated.

Besides the “Component Name” and “State Chain Type” attributes, each record in the grid has both “PBS Code” (Product Breakdown Structure Code) and “Parent Code” attributes. These two codes define the parent-child relationships between the product components (tree nodes) in the product decomposition hierarchy.

The “PBS Code” is the unique identifier to locate a product component in the database, while the “Parent Code” is actually the “PBS Code” of the parent component. The product component grid in Figure 6.7, for example, shows that the “Parent Code” of the deck segment “SegL13” is “01010100”, which is the “PBS Code” of the subsystem component “Left Deck System”, indicating that “Left Deck System” comprises the “SegL13” deck segment as its child components.

The application automatically generates the state chain according to the “State Chain Type” attribute of each product component, using the “Initialize Chain” button in the navigation tool bar in the bottom left panel of the interface as shown in Figure 6.7. The Figure shows the automatically generated chain for “SegL13” comprising a sequence of six component states. Additionally, the component state grid in the bottom left of Figure 6.7 shows that the “StartD”, “ActiveFinishD”, and “FinishD” attributes are assigned a default time “0”, indicating that these temporal attributes have not been computed by the application. These three temporal attributes are not represented in the format of date time, but in the format of the number of days from the start of the study period, which is similar to the representation format of “StartD” and “FinishD” attributes in Figure 6.5.

The “State Chain Type” is managed through the state chain type browser shown in Figure 6.8. The browser manages the intermediate function knowledge with respect to state chain type. The state chain type for a product component is selected based on the construction method. Figure 6.8 shows the state chain of “Deck Segment”, which was replicated in Figure 6.7 for initializing the construction life cycle of the deck segments.

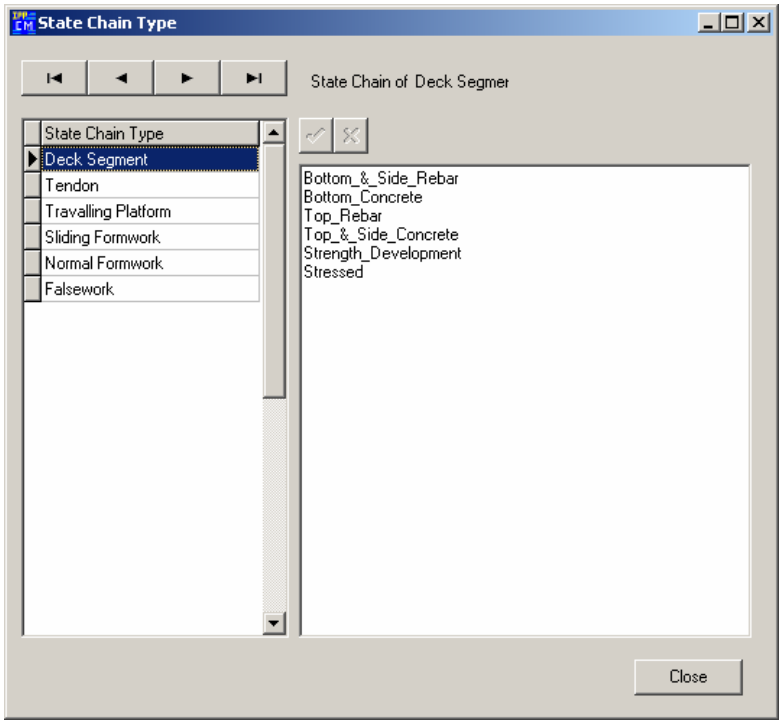



**Figure 6.8 State Chain Type Browser**

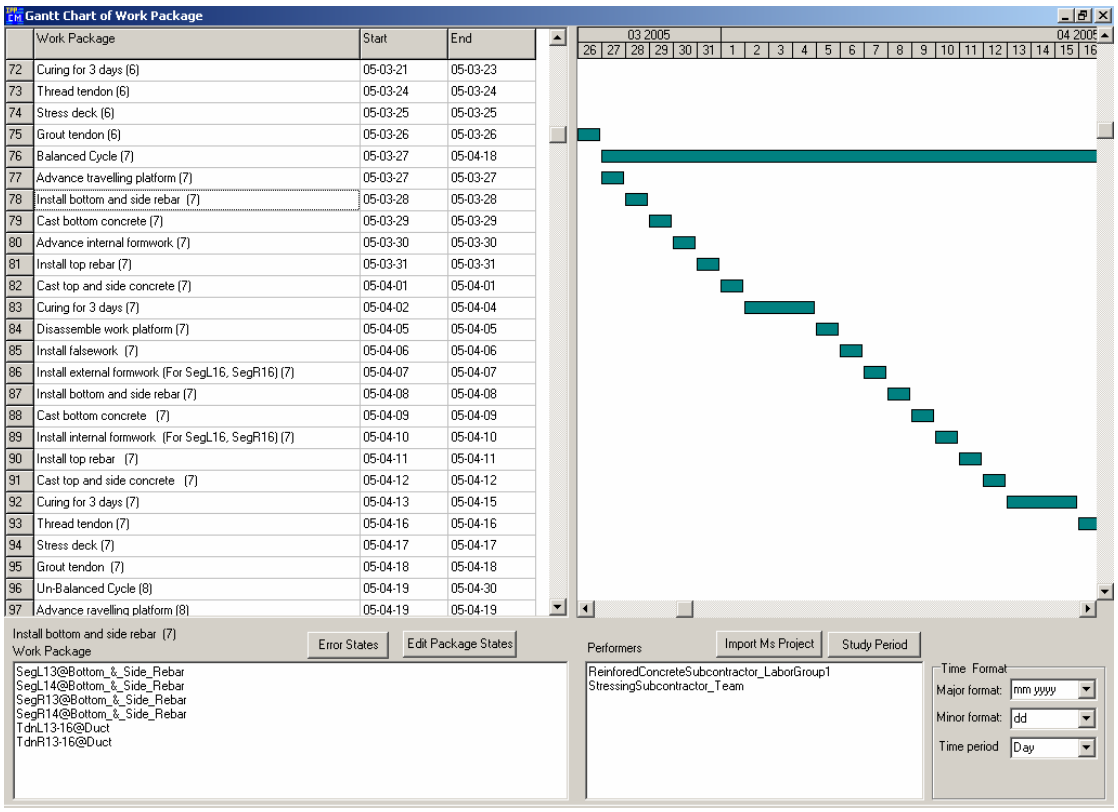



**Figure 6.9 Work Package and Performers of Construction Processes**

After the process schedule has been imported and the state chain of each product component defined, the work packages of the construction processes are

defined using the interface depicted in Figure 6.9. Simultaneously, the performers of the construction processes can also be entered. The one-to-many relationships between construction processes and performers are stored in the process schedule database, and the sequence of performer states of a performer can hence be derived as elaborated in the preceding sections.

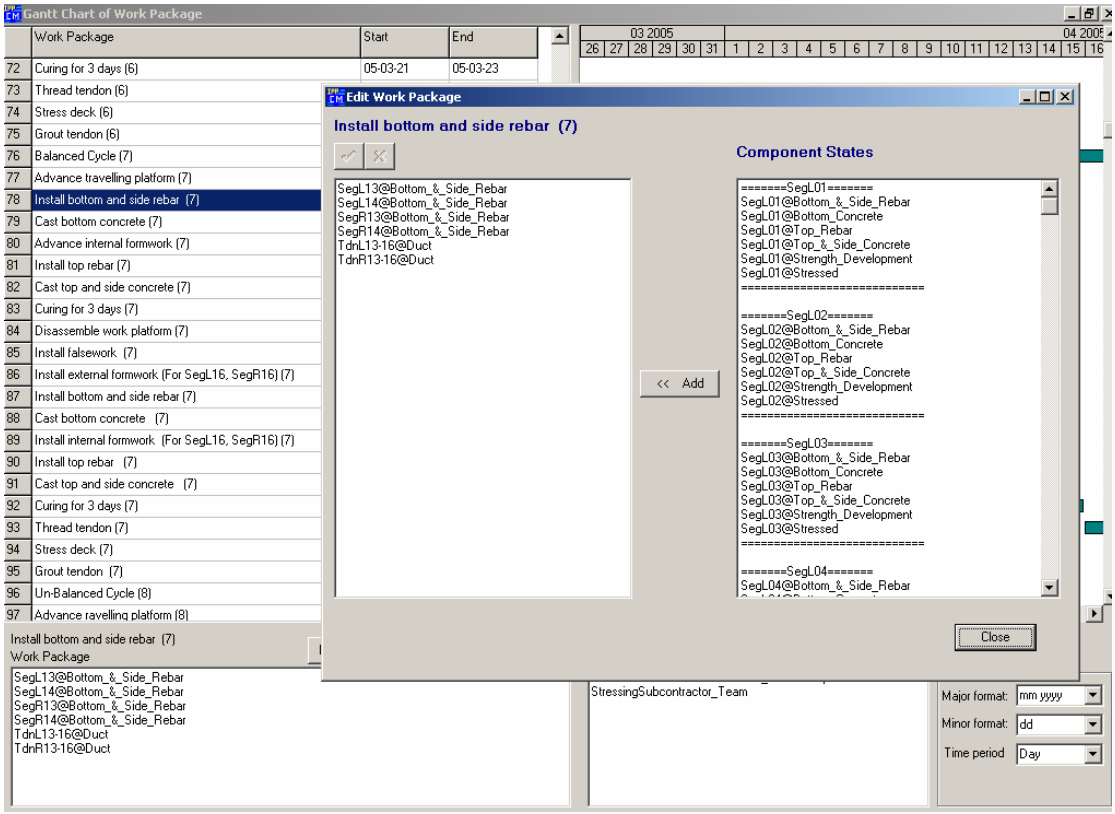



**Figure 6.10 "Edit Work Package" Window**

The input of work package is facilitated with the "Edit Work Package" window as shown in Figure 6.10. The edit box in the right side displays all the component states, which can be selected and added to the work package box on the left side. As demonstrated in Figure 6.10, the six component states "SegL13@Bottom_&_Side_Rebar", "SegL14@Bottom_&_Side_Rebar", "SegR13 @Bottom_&_Side_Rebar", "SegR14@Bottom_&_Side_Rebar", "TdnL13-L16@ Duct", and "TndR13-R16@Duct" are added from the candidate component state list on the right and associated with work package of the construction activity "Install

Bottom and Side Rebar (7)". In this way, the work packages bridge the construction activities in the process schedule and the component states in the in-progress product model. Consequently, the "StartD", "ActiveFinishD", and "FinishD" temporal attributes of the associated component states can be automatically computed through the corresponding work packages.

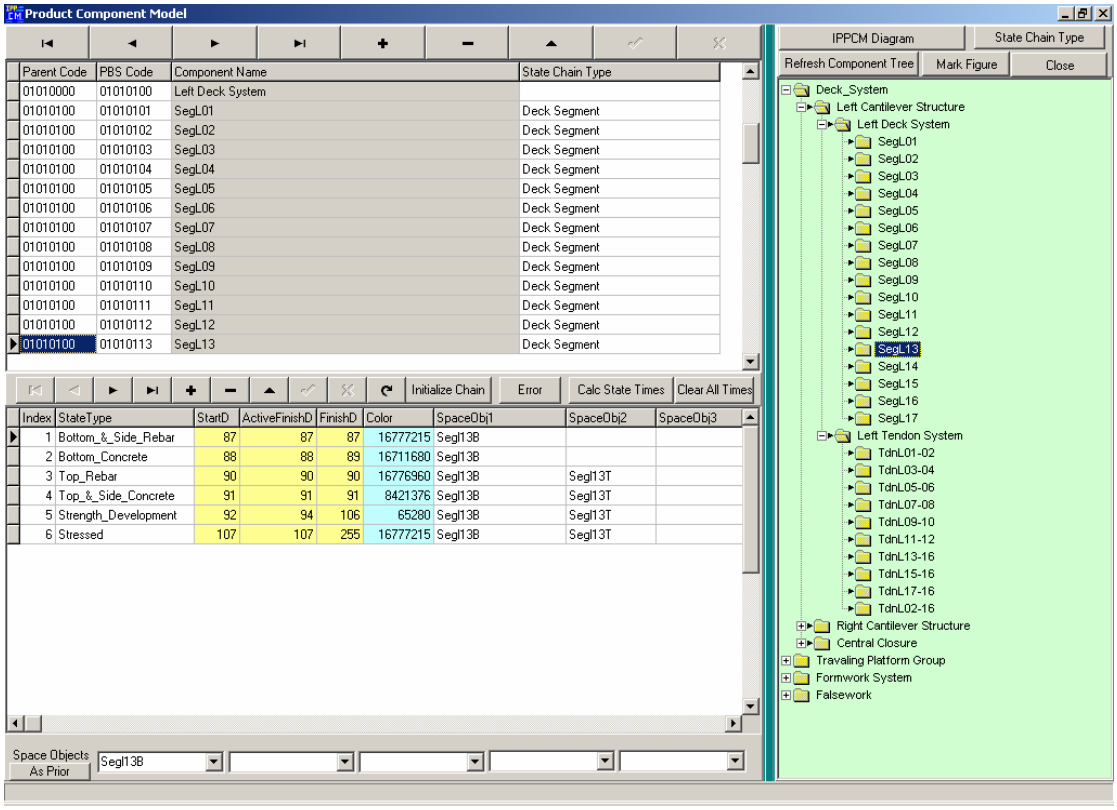



**Figure 6.11 Automatic Computation of Temporal Attributes of States**

Figure 6.11 shows that the three temporal attributes of the state "Bottom & Side Rebar" of "SegL13" are on Day 87, indicating that this state has the active phase of 1 day and zero-duration quiescent phase. This computation resultant is consistent with the process schedule as shown in Figure 6.9 where the activity "Install Bottom and Side Rebar (7)" starts and finishes on March 28$^{th}$, 2005. There are exactly 87 days from March 28$^{th}$, 2005 backward to January 1$^{st}$, 2005 (the start of the study period as shown in Figure 6.6). The activity "Install Bottom and Side Rebar (7)" is immediately followed by the activity "Cast Bottom Concrete (7)" (starting on March 29$^{th}$, 2005) so

that the state “Bottom & Side Rebar” of “SegL13” has no quiescent duration, and its “FinshD” attribute is determined by the “StartD” of “Cast Bottom Concrete (7)”. Meanwhile, The “StartD” and “ActiveFinishD” temporal attributes of the “Duct” states of the tendon component “TndL13-L16” are respectively equal to the “StartD” and “FinishD” of “Install bottom and side Rebar” since it is comprised in its work package.

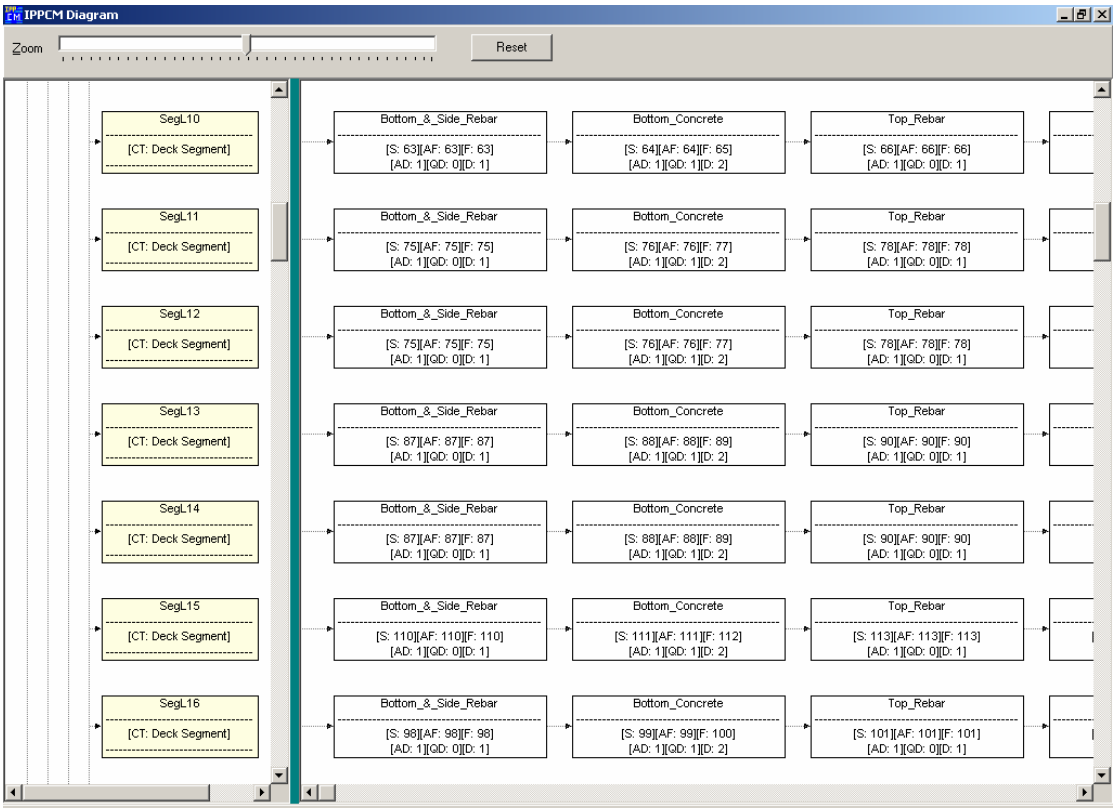



**Figure 6.12 Box View of Component Construction Life**

Figure 6.12 illustrates the box view of the construction life cycle of the deck components “SegL10” to “SegL16”. On the left side of the interface is the product decomposition tree, while on the right is the component state chain for depicting the construction life cycles of the deck segments on the left side. Each box in the network represents a component state. Besides the state type attribute, a state box also displays five temporal attributes: “StartD” (“[S: ]”), “ActiveFinishD” (“[AF: ]”), “FinishD” (“[F: ]”), “Active Duration” (“[AD: ]”), and “Quiescent Duration” (“[Q: ]”). In this

way, the planning engineers can have an overview of the in-progress configuration of a facility product.

### 6.7.2 4D Simulation Interface

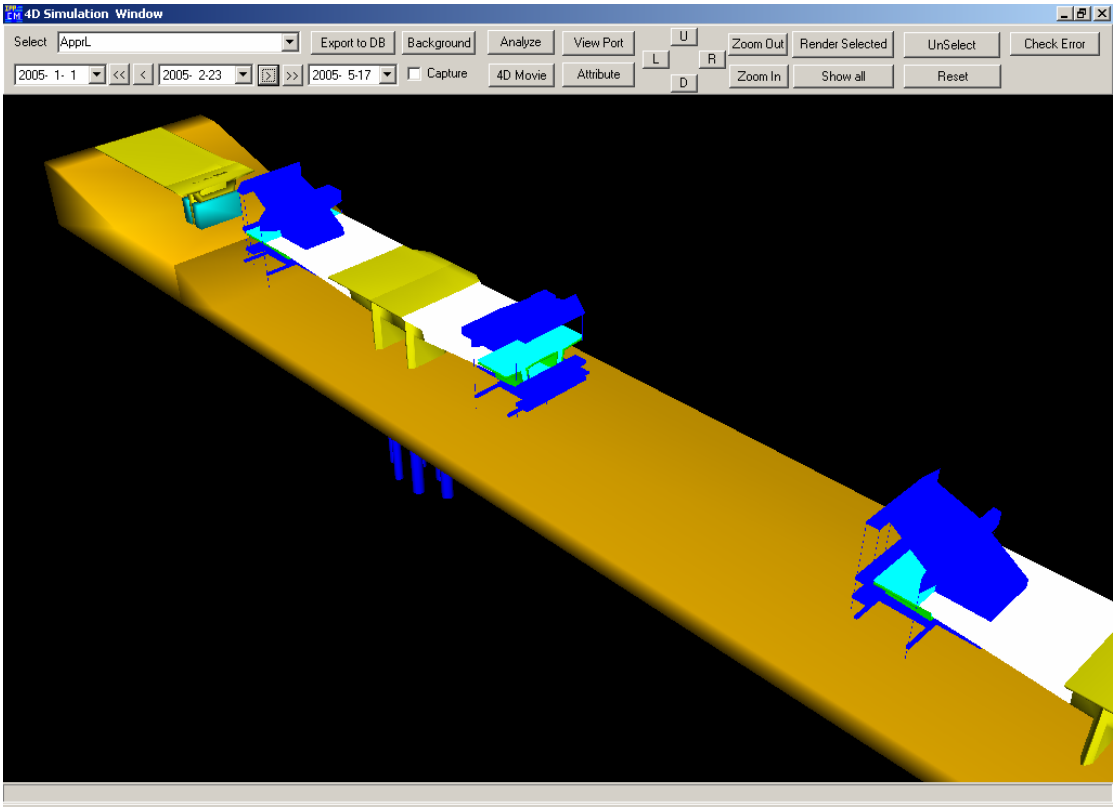
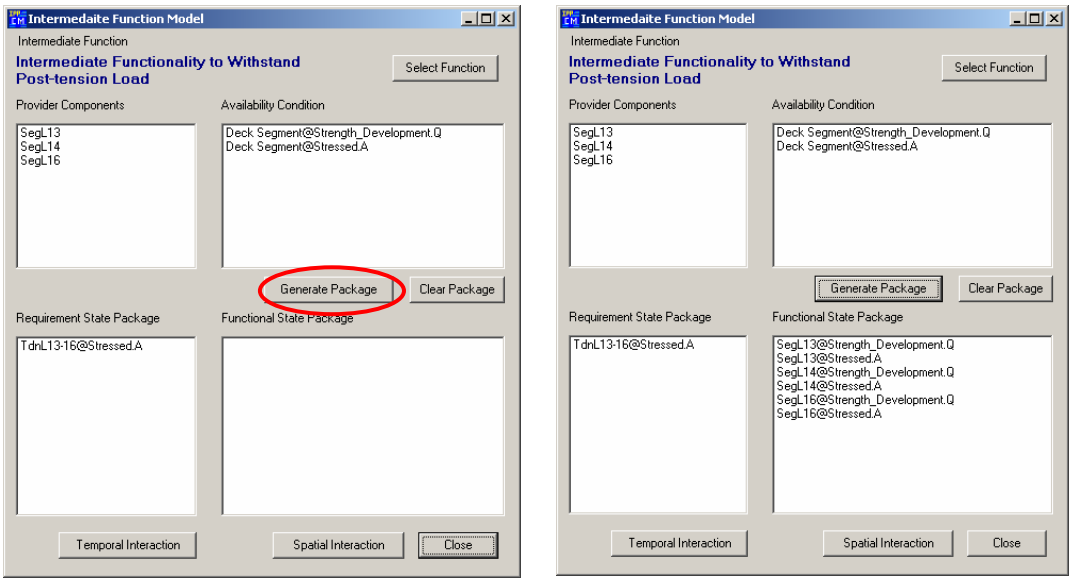



**Figure 6.13 4D Simulation of Deck Construction**

Figure 6.13 shows the 4D simulation interface that visualizes the construction schedule of constructing the bridge deck through its in-progress product model. The 4D simulation engine traverses the collection of physical components (including both product and resource component) to "filter" the product components via their existence vectors. These filtered product components are being constructed at the time of the present frame marker, and their physical shapes should be visible on the screen. The rendering color of the component is determined by its state at the time of the simulation frame. Each frame can be captured and recorded as a picture, and such a sequence of pictures can then be labeled and composed into a 4D movie.

### 6.7.3 Intermediate Function Analysis Interface




**A** **B**

**Figure 6.14 Intermediate Function Browser**

Figure 6.14 shows the browser window for entering the attributes of an intermediate function object. In Figure 6.14A, the requirement state package, the provider component and the availability condition of the "Intermediate Functionality to Withstand Post-tension Load" have been entered, but the functional state package remains null. Via the "Generate Package" button, the "Functional State Package" is automatically filled in by the inference engine. In contrast, the requirement state package requires the manual input of the user.

Figure 6.14B shows altogether six functional states for three provider components that are automatically generated. Specifically, each of the components "SegL13", "SegL14", and "SegL16" has two functional state phases "Strength Development.Q" and "Stressed.A", as defined in the "Availability Condition". This means that the three deck segments (provider components), during these two state phases, can withstand the post-tension load exerted by the user component "TdnL13-L16" when it is being stressed (during the requirement state "TdnL13-

L16@Stressed.A"). These six generated functional states will be used by the inference engine to compute the availability time-window for evaluating the fulfillment of the intermediate function requirement.

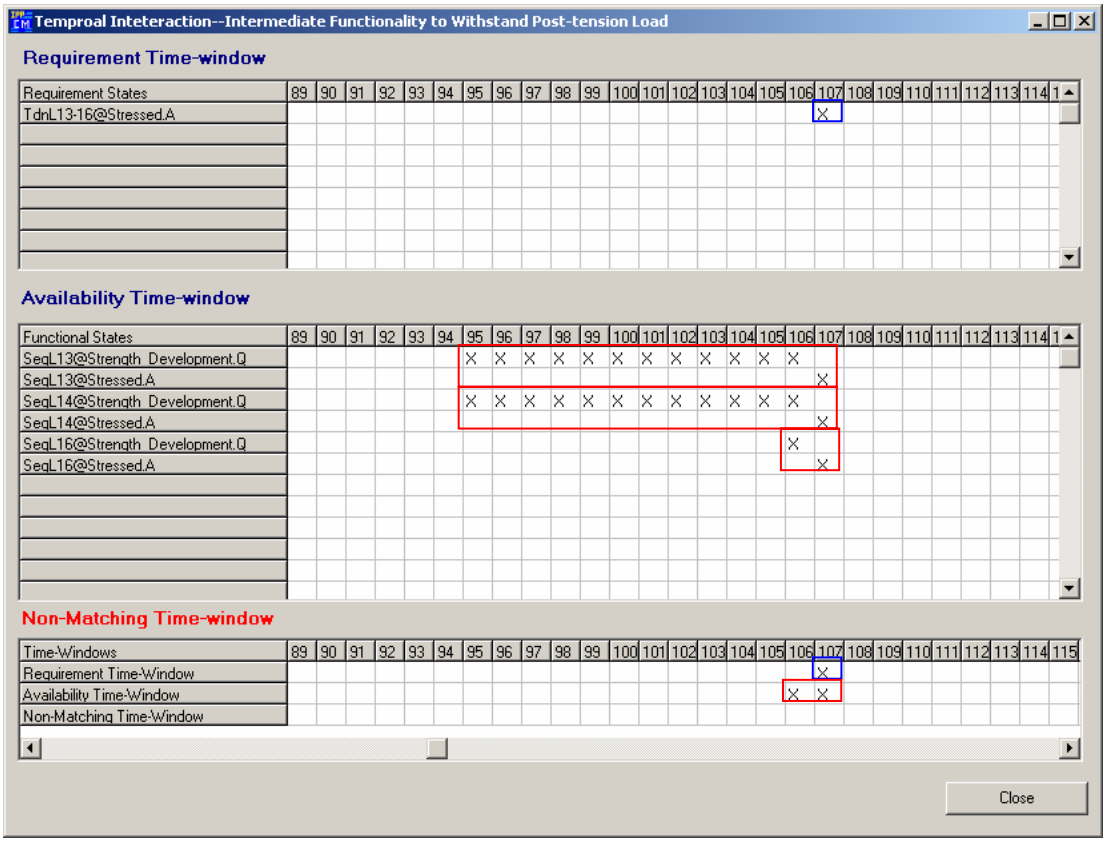



**Figure 6.15 Interface of Temporal Interaction Analysis**

The "Temporal Interaction" function on the bottom left of the browser shown in Figure 6.14 brings up the temporal interaction analysis interface as depicted in Figure 6.15. This interface publishes the results of the temporal interaction analysis for "Intermediate Functionality to Withstand Post-tension Load". There are three grids for displaying the requirement time-window, availability time-window, and non-matching time-window, respectively. The number in the top row of each grid denotes the number of the finite interval unit of the existence vector. In the "Requirement Time-Window" grid, each requirement state as well as its existence vector is displayed. Similarly, each functional state as well as its existence vector is

demonstrated in the “Availability Time-Window” grid. The functional states are indexed by the component name as shown in Figure 6.15. The “Non-Matching Time-Window” grid shows the existence vectors of the requirement, availability, and non-matching windows in three rows. In this way, the availability of the temporal interaction between the function user and the function provider can be evaluated using the equations Eq.’s (6.1) to (6.5). Figure 6.15 shows that the non-matching time-window is “null”. This indicates that the “withstanding post-tension load” functionality is provided by the three deck segments (“SegL13”, “SegL14”, and “SegL16”) when the tendon “TdnL13-16” is being stressed.

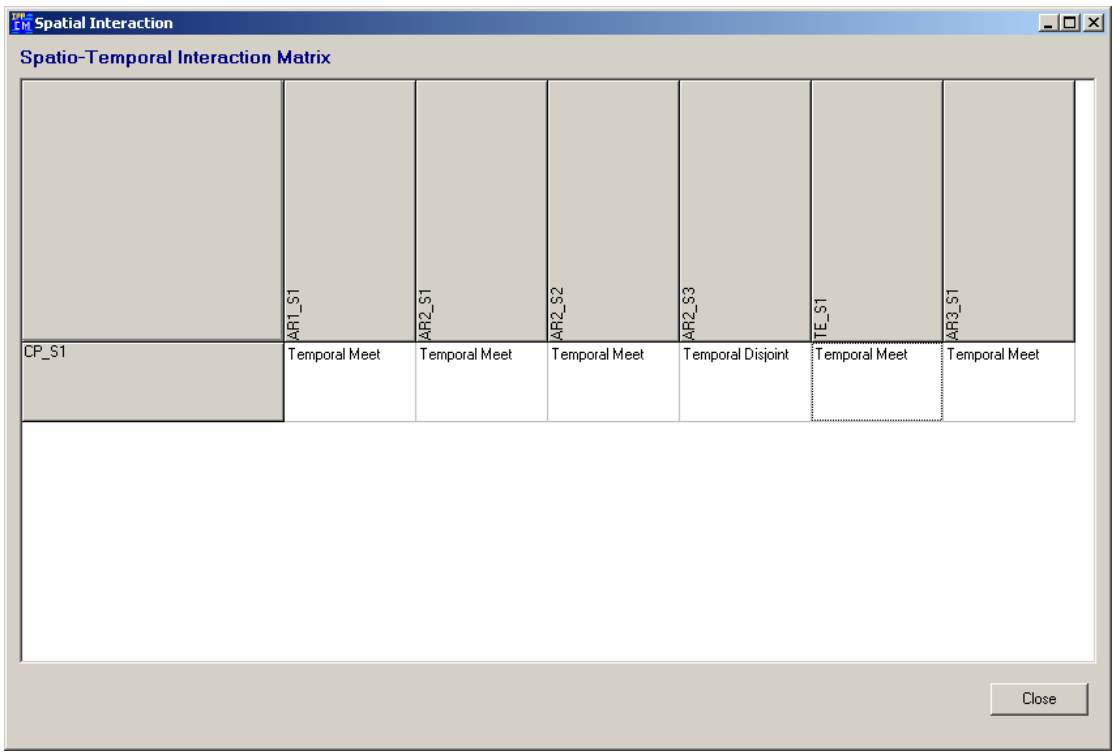



**Figure 6.16 Interface for Publishing Spatio-Temporal Interaction Matrix**

Figure 6.16 depicts the interface of the Spatio-Temporal Interaction Matrix for displaying the spatial interaction between the function user and the function provider. Each row title of the matrix grid denotes a temporal space entity referenced by the corresponding requirement state, while each column title denotes a temporal space entities referenced by the corresponding functional state. The other grid cells show the

binary temporal topological relationships between the corresponding row and column entities.

The interface shows the computation result of the example of moving a mobile crane on an excavated road as elaborated in Section 5.2.3. The mobile crane (user component) references only one space entity “CP_S1” which depicts the path space of a mobile crane, while three access road segments (provider component) reference a total of six space entities. Consequently, the temporal topological relationships between the space entity defining the path space of the mobile crane and the six space entities depicting the access road segments are respectively computed by the inference engine. The highlighted cell demonstrates an undesirable temporal topological relationship, indicating that the path space “CP_S1” of the mobile crane should but cannot temporally “meet” the shape entity “TE_S1” due to the trench excavation. In this way, the unavailable spatial interaction between the “Mobile Crane” and the “Access Road” can be identified.

## 6.8 Concluding Remarks

This chapter presents the research prototype 4D intermediate Function AnalysiS Tool (4D-iFAST) to implement the developed information integration framework and the analysis methodologies as well as 4D simulation. The main data structure of the prototype is represented, and the mechanisms of the inference engine and the 4D simulation engine are also addressed. Lastly, some typical interface windows are demonstrated for illustrating the key functions of the prototype. These research results indicate the intermediate function analysis framework can be implemented as a software product, taking the advantages of information technologies for enhancing construction requirement management.

Particularly, the existence vector concept and the corresponding Boolean operations are developed for implementing the temporal and spatial interaction analysis. In this way, the temporal and spatial interactions can be evaluated without the need to conduct simulation of construction schedules. Additionally, the temporal data representation in intermediate function analysis can be simplified by using existence vector to represent both continuous temporal intervals and discontinuous time-windows, and the interval algebra computations are also speed up by using the Boolean operations on existence vectors. Meanwhile, the 4D simulation can facilitate intermediate function analysis with respect to identifying function user and providers and to locating alternative engineering solutions for better realizing intermediate functionalities as well as to identifying state relationships.

# CHAPTER 7 CASE STUDIES

This chapter addresses two case studies for illustrating the application of the developed concepts, representation schemata, integration framework, and analysis tools. These two case studies come from the real projects and are intentionally amended to keep confidential some sensitive data, but the characteristics of associated intermediate function analysis are kept as original. The first case study is used to demonstrate the representation of intermediate function knowledge, and the identification and analysis of the bottleneck state as well as 4D simulation of construction schedules. The second case is employed to demonstrate the analysis of co-matching the requirement time-windows and the availability time-windows of two compatible intermediate functionalities through the collaboration among the subcontractors, which can be facilitated by the spatio-temporal enquiry on the 4D model and.

## 7.1 Case 1: Post-Tensioned Prestress Bridge by Balance Cantilever Approach

### 7.1.1 Balance Cantilever Construction Approach

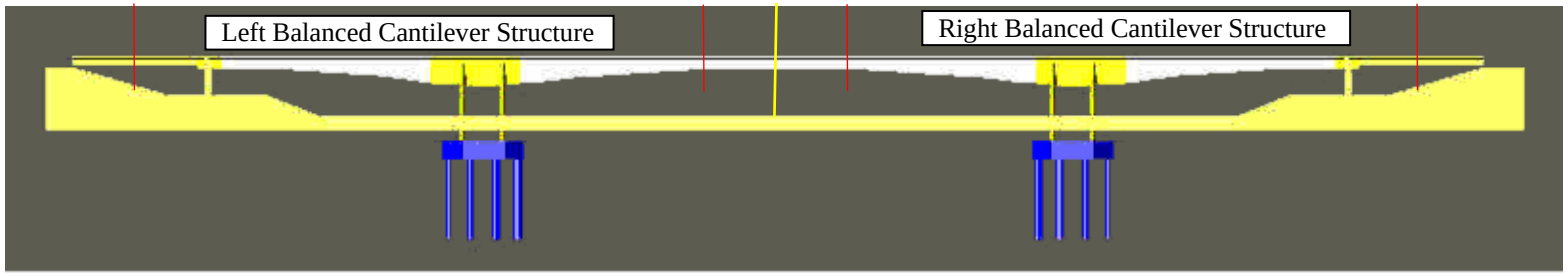



**Figure 7.1 Symmetric Structure of Bridge**

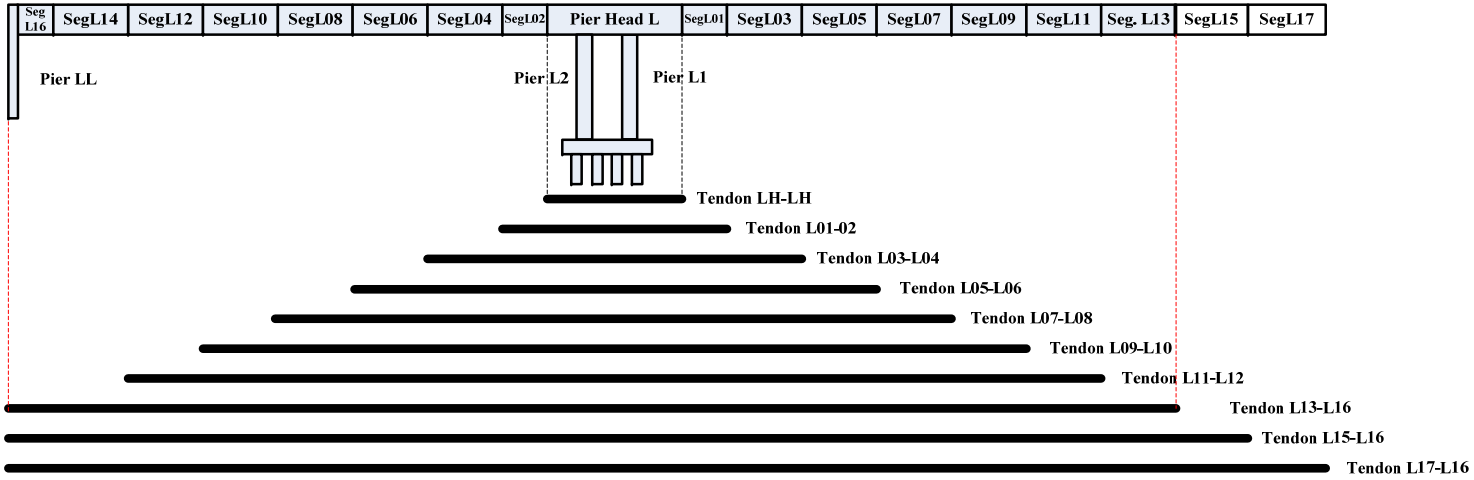



**Figure 7.2 Left Balanced Cantilever with Tendon Configuration**

The following case involves the construction of the deck of a post-tensioned prestress bridge across a river (Figure 7.1) using the balance cantilever approach. The balance cantilever approach from two piers was adopted because the width of the river makes crane lifting method infeasible. Accordingly, two traveling platforms were used to support the construction of the bridge deck segments on each side of the piers as construction proceeds. Additionally, the bridge structure is symmetric from the mid-span (see Figure 7.1), and the construction of the right balanced cantilever structure follows a similar construction sequence for constructing the left balanced cantilever structure, so only the left cantilever is elaborated in this case study. Furthermore, Figure 7.2 shows the deck segments numbering together with the prestress tendon configuration.

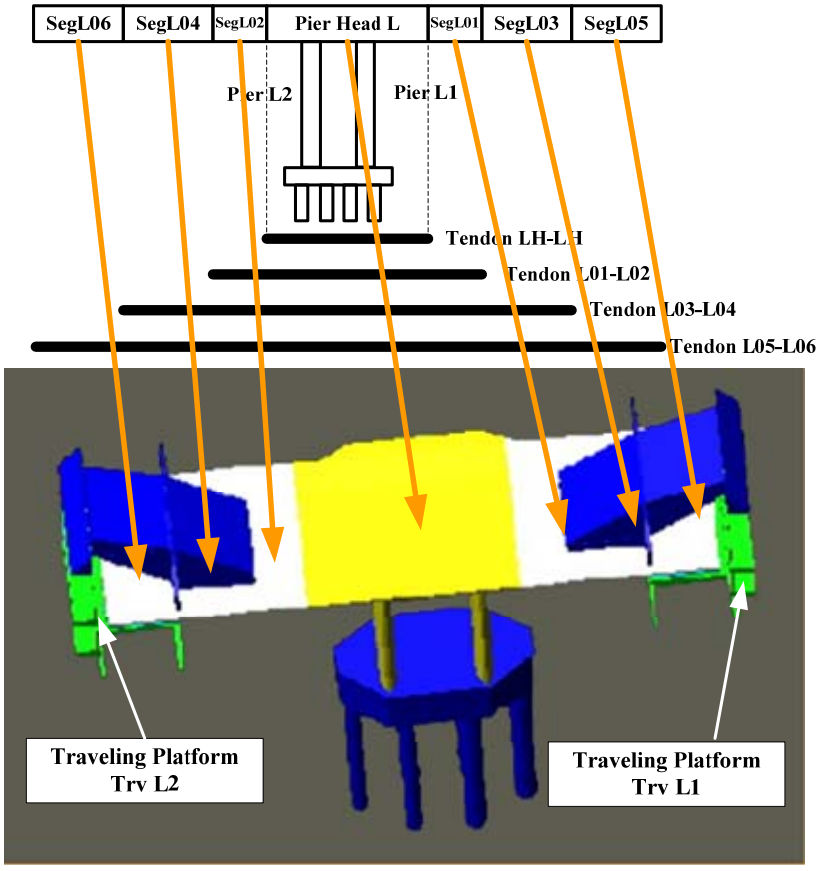



**Figure 7.3 Balanced Cantilever Structure With Traveling Platforms in Cycle(3)**

The two cantilevers on each side of the pier are constructed outward from the pier with not more than one segment out of balance at any time so that the stability of the in-progress structure is maintained. For example, Figure 7.3 shows a frame from the 4D simulation of the deck construction, either SegL05 or SegL06 cannot be constructed until both SegL03 and SegL04 have been stressed to form a stable balanced cantilever in the construction cycle "Cycle(2)". The construction consequence of the bridge structure is illustrated with 4D simulation frames in the succeeding section.

### 7.1.2 4D Simulation of Original Construction Sequence

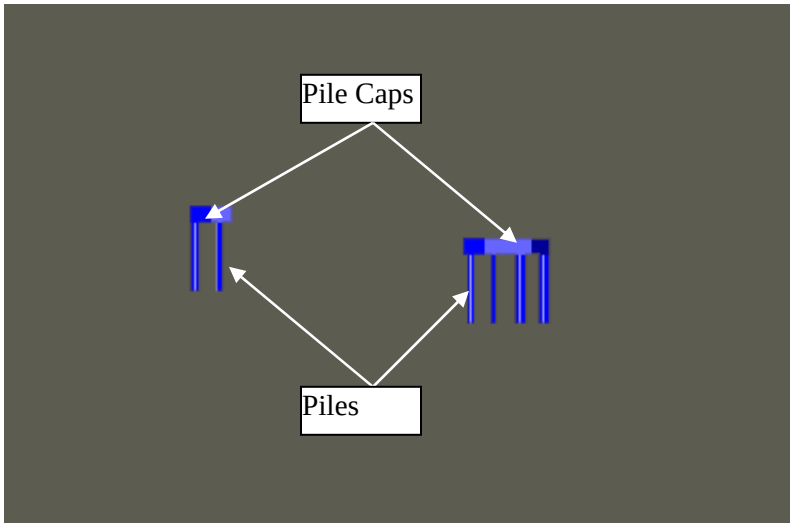



**Figure 7.4 Construction of Pile Foundation System**

The general construction sequence of the bridge can be separated into three sequential phases. During the first phase, the piles are firstly installed and driven into the river bed and banks, and then the pile caps are built to provide the foundation systems for the upper piers and the deck (Figure 7.4). Subsequently, the "Piers LL, L01, and L02" are constructed on the pile caps, and then the pier caps are constructed and post-tensioned

with the temporary support functionality provided by the falseworks at Piers LL, L01, L02 (Figure 7.5).

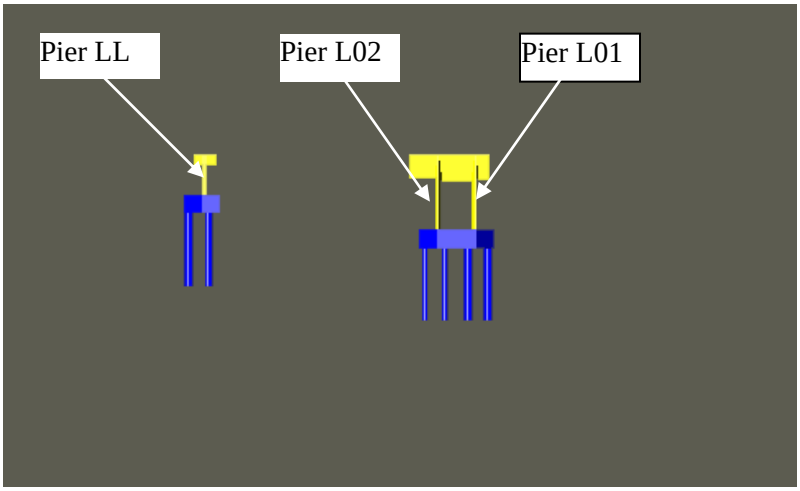



**Figure 7.5 Construction of Piers**

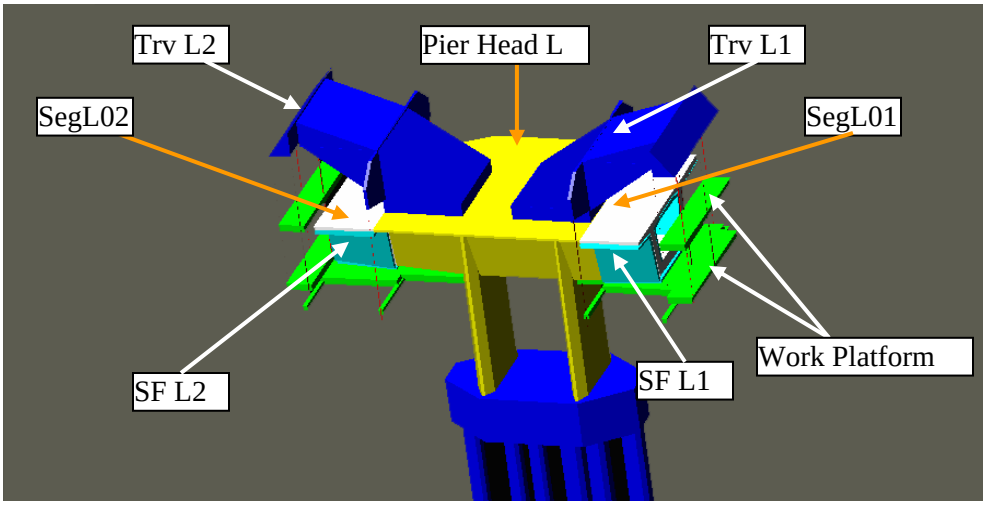



**Figure 7.6 First Cycle of Balanced Cantilever Construction**

During the second phase, the deck system is sequentially constructed as illustrated by Figures 7.6 to 7.10. The traveling platform “Trv L1” will be first installed on the completed pier head “Pier Head L” (see Figures 7.6), and then the sliding formwork “SF L1” will be installed on the traveling platform “Trv L1” for constructing the deck segments. Using the temporary support provided by the traveling platform “Trv L1” and

the sliding formwork “SF L1” on it, the deck segment “SegL01” can be constructed by installing rebar and casting concrete as well as developing strength. Similarly, the deck segment “SegL02” can be constructed using the temporary support provided by the traveling platform “Trv L2” and the corresponding sliding formwork “SF L2”. Additionally, Figure 7.6 also illustrates that each traveling platform has a two-level work platform, which is required for supporting the workers and the post-tensioning equipment. These two segments can be post-tensioned with the pier head “Pier Head L” to form a temporary cantilever structure after they have been cast and cured for three days. Subsequently, the tendon ducts should be grouted for protecting the tendon clusters. This is the first cycle of the balanced cantilever construction.

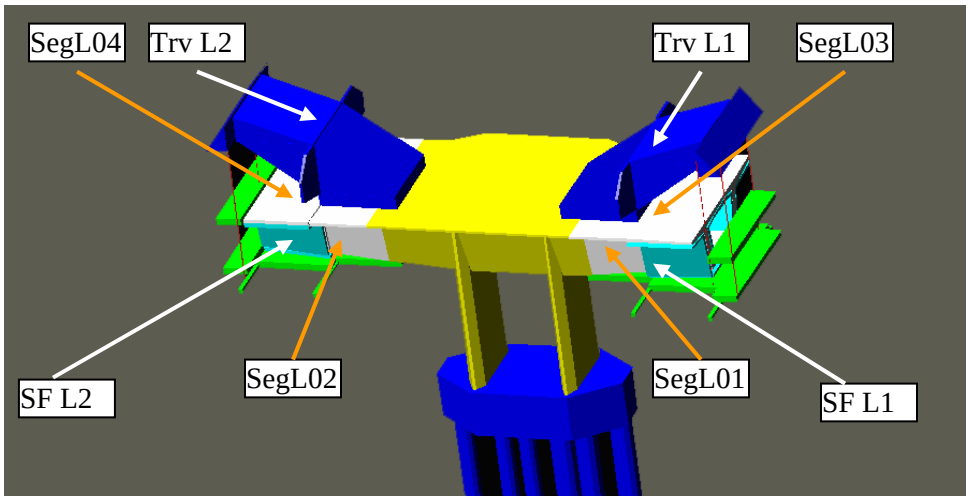



**Figure 7.7 Construction of Deck Segments “SegL03” and “SegL04”**

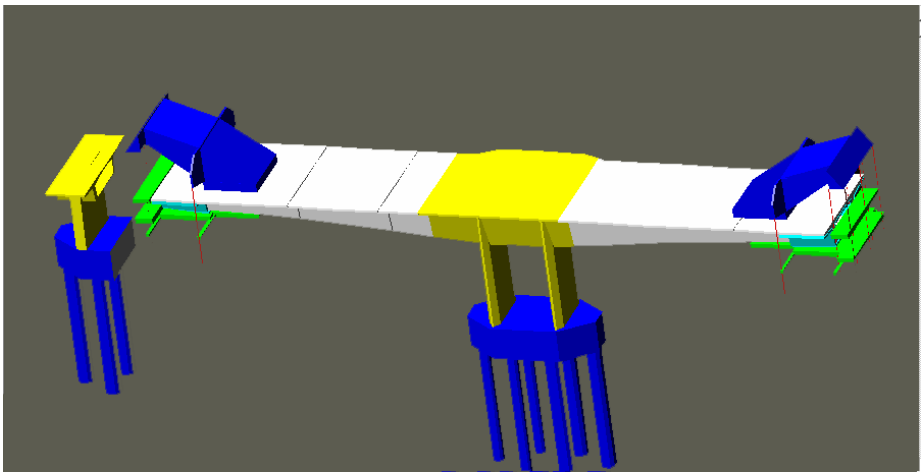


**Figure 7.8 Construction of Deck Segments "SegL11" and "SegL12"**

In the end of the first construction cycle, the in-progress cantilever structure can provide temporary support functionality to permit the traveling platforms "Trv L1" and "Trv L2" to be advanced outward from the pier system in order to construct the deck segments "SegL03" and "SegL04" in the second construction cycle. Figure 7.7 shows the simulation frame with the sliding formworks "SF L1" and "SF L2" having been advanced outwards since the deck segments "SegL01" and "SegL02" are self supported. The above-mentioned balance cantilever construction procedure will be repeated through "SegL05" to "SegL12". The construction of "SegL11" and "SegL12" in the sixth construction cycle is shown in Figure 7.8.

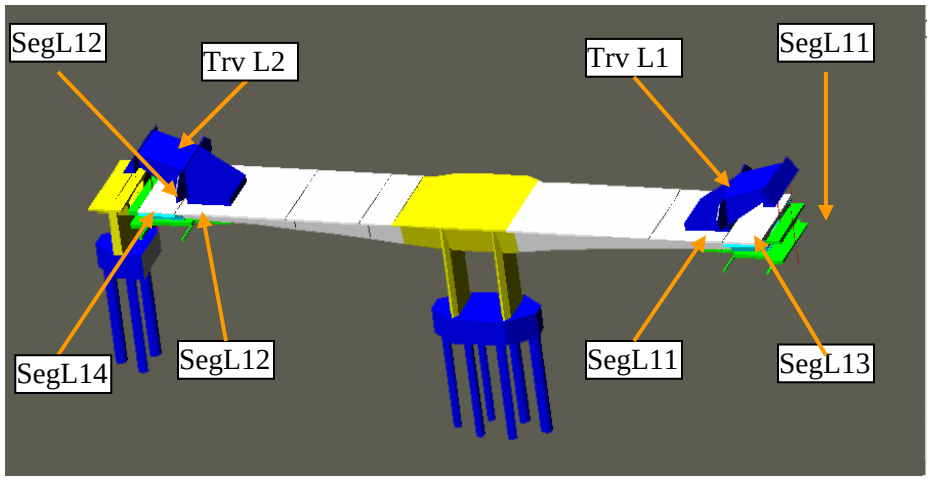



**Figure 7.9 Construction of Deck Segments "SegL13" and "SegL14"**

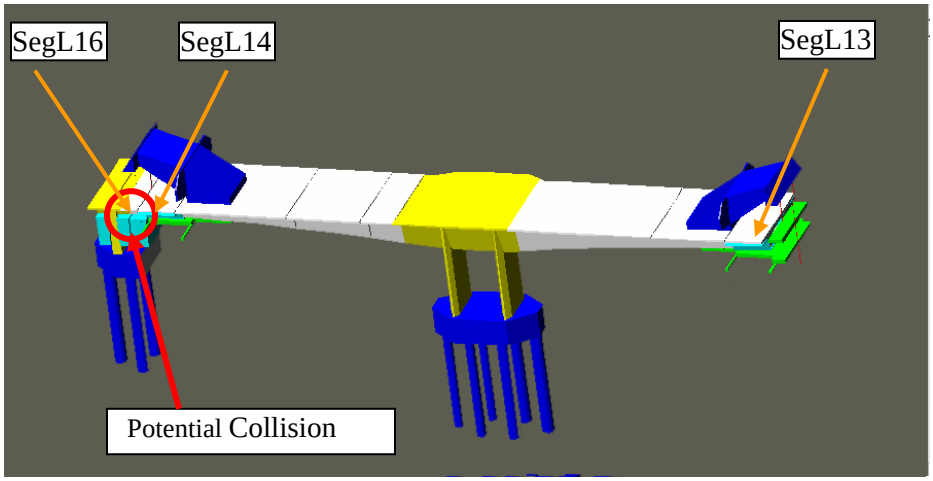



**Figure 7.10 Construction of Deck Segment "SegL16"**

After grouting the duct of "Tendon L11-L12" that stressed "SegL11" and "SegL12" with the in-progress cantilever structure, the traveling platform "Trv L1" and "Trv L2" will be advanced and fixed onto the "SegL11" and "SegL12", respectively, which starts the seventh and last cycle to construct the balance cantilever structure (Figure 7.9). The original method of construction stipulates that the construction of the deck segments "SegL13" and "SegL14" is accomplished using the traveling platforms "Trv L1" and "Trv L2", respectively (Figure 7.9), and this is followed by the construction of the closure segment "SegL16" with the support of the falsework "FWL16" instead of

the traveling platform “Trv L2” (Figure 7.11). Subsequently, three deck segments SegL13, SegL14 and SegL16 will be stressed together via “TendonL13-16” as shown in the tendon configuration of Figure 7.2.

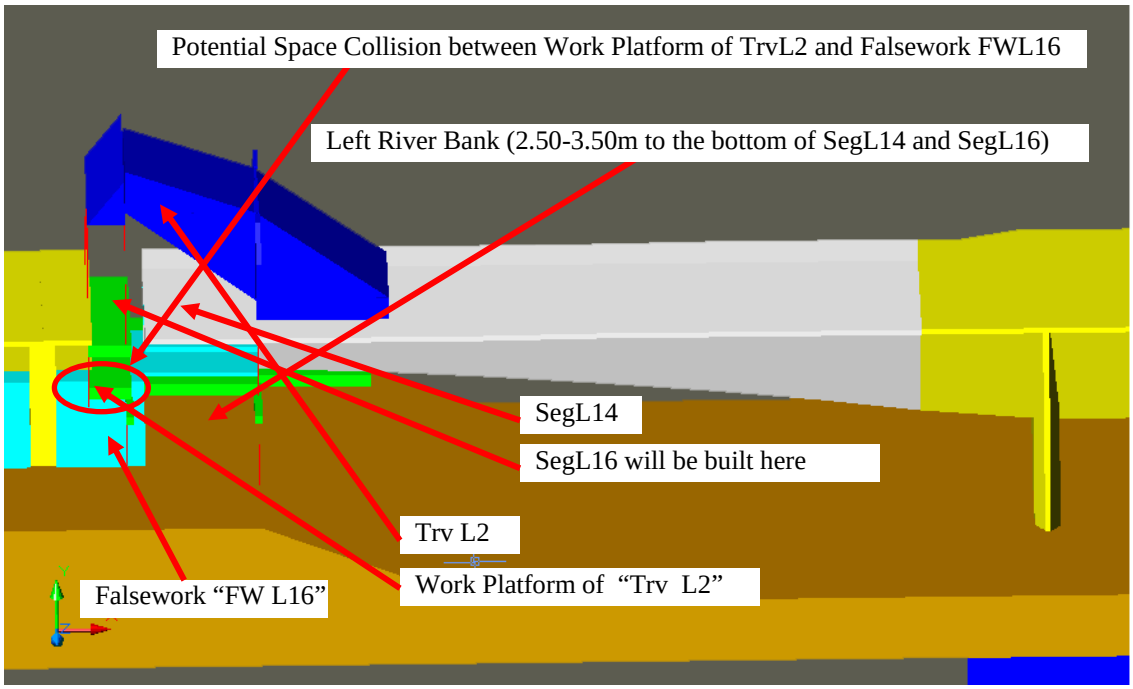



**Figure 7.11 Potential Collision between Work Platform and Falsework**

The simulation frame shown in Figure 7.10 also implies a potential space collision. The potential collision is zoomed in a larger viewport as shown in Figure 7.11. In order to prevent the potential collision, the falsework “FW L16” under the deck segment “SegL16” should be erected after the work platform of “Trv L2” (in the location of SegL16) is disassembled. Otherwise, a space collision between the falsework “FW L16” and the work platform of “Trv L2” may occur. This implies that the construction of “SegL14” and “SegL16” in the same construction cycle cannot commence simultaneously due to the potential space utilization conflicts.

The third construction phase of the deck structure begins after “Tendon L13-L16” has been stressed and grouted. “SegL15” and “SegL17” are sequentially constructed in an

unbalanced manner (Figure 7.1). After that, the middle closure between the left and right deck structures can be constructed (Figure 7.1), and then the continuity tendons “Tendon L16-L17” (See Figure 7.2) going through the two end spans and the middle span will be stressed to make the whole deck system as an integrity structure.

### 7.1.3 Intermediate Function Requirement Knowledge Representation

Figures 7.12 to 7.21 illustrate the three types of intermediate function requirement knowledge capturing the temporal logics that reside in this construction approach.


**Type Name:** Deck_Segment_Chain
**Component Type:** Deck_Segment
**Construction Method:** Cast-in-Situ
**State Chain:** {Bottom_&_Side_Rebar; Bottom_Concrete; Top_Rebar; Top_&_Side_Concrete (A); Strength_Development; Stressed.}

**Figure 7.12 State Chain Type of Deck Segment**

Figure 7.12 shows the state chain type “Deck Segment Chain”, representing the construction life cycle of product component type “Deck Segment”. It depicts that a typical deck segment is constructed from the bottom to top using the cast-in-situ construction method. Specifically, its construction life cycle comprises six sequential state types, namely “Bottom and Side Rebar”, “Bottom Concrete”, “Top Rebar”, “Top and Side Concrete”, “Strength Development (for stressing)”, and “Stressed” (see Figure 7.13). Additionally, the boundary shape of the deck segment associated with each of the states along the construction life cycle is shown in the bottom of Figure 7.13.

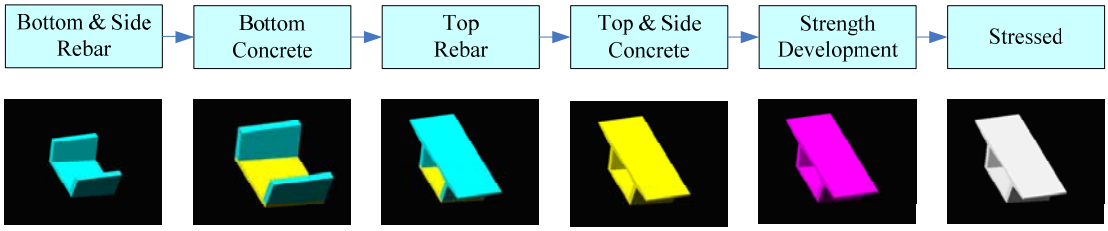
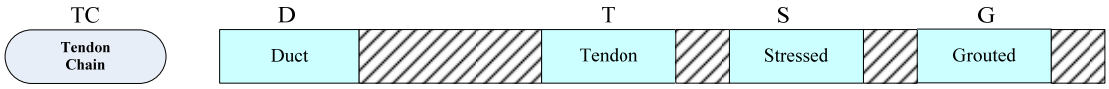



**Figure 7.13 State Chain of Deck Segment Component with State Spaces**

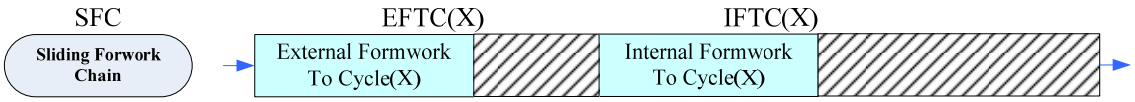
This sequence not only depicts the gradual development of material composition in the evolution of the construction cycle but also indicates the transitions of engineering behavior characteristics. The first four states correspond to changes in material composition, wherein the rebar is assembled in the respective slabs and sides, and the concrete cast. The latter two states correspond to changes in strength. Additionally, the state "Top & Side Concrete" have no quiescent phase, since the active "Strength Development" state, which is induced by the hydration process, starts immediately after the concrete has been cast.




**Type Name:** Tendon_Chain
**Component Type:** Tendon
**Construction Method:** Balanced Cantilever Construction Method
**State Chain:** {Duct; Tendon; Stressed; Grouted.}

**Figure 7.14 State Chain Type of Tendon**

The "Tendon Chain" in Figure 7.14 describes the construction life cycle of the "Tendon" component for stressing the deck segments into an integrated structure. It comprises the "Duct" state when the duct is installed within the side rebar cage of the deck segment, the "Tendon" state when the tendon clusters are threaded, the "Stressed" state when the tendons are stressed, and the "Grouted" state when the tendon duct is grouted with mortar to protect the stressed tendons.




**Type Name:** Sliding_Formwork_Chain
**Component Type:** Sliding_Formwork
**Construction Method:** Balanced Cantilever Construction Method
**State Chain:** {External Formwork To Cycle(X); Internal Formwork To Cycle(X).}

**Figure 7.15 State Chain Type of Sliding Formwork**

The sliding formwork system contains two parts: “External Formwork” and “Internal Formwork”. Its construction life cycle simply comprises two sequential state types, namely “External Formwork To Cycle(X)” and “Internal Formwork To Cycle(X)”, which represent the sliding of the respective formwork parts to their locations for supporting the deck segment that will be constructed in “Cycle(X)” (see Figure 7.15). The “X” in “Cycle(X)” herein indicates the order number of the construction cycle that the traveling platform is planned to support. These two states will be alternated as the sliding formwork is advanced forward by the traveling platform to its new locations in the succeeding cycles.


**Type Name:** Traveling_Platform_Chain
**Component Type:** Traveling_Platform
**Construction Method:** Balanced Cantilever Construction Method
**State Chain:** {…; Platform To Cycle(X);...}

**Figure 7.16 State Chain Type of Traveling Platform**

Figure 7.16 shows the state transition life cycle of a traveling platform used in the balance cantilever construction approach. The states of the traveling platform are characterized by its locations, since its support functionality is only accessible within the

boundary of the platform. The “Traveling Platform” is moved to its location for “Cycle(X)” in order to support the weights of the sliding formwork and the deck segment as well as other construction loads. Accordingly, the states of the traveling platform are denoted by the construction cycle “Cycle(X)”.

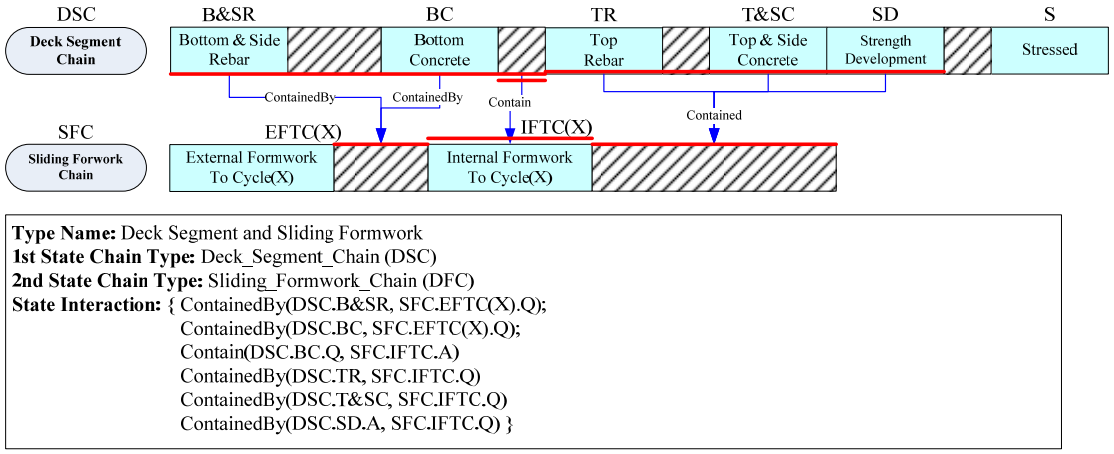



**Figure 7.17 State Interaction Type between Deck Segment and Sliding Formwork**

Figure 7.17 illustrates the state interaction knowledge to depict the functional interdependencies between a deck segment and the corresponding sliding formwork (supported by the traveling platform). The construction life cycles of these two component types are respectively defined by “Deck Segment Chain” (first state chain) and “Sliding Formwork Chain” (second state chain).

There are altogether six functional interdependencies between the above-mentioned state chains, which are grouped in three interaction sets. The first set comprising “ContainedBy(DSC.B&SR, SFC.EFTC(X).Q)” and “ContainedBy(DSC.BC, SFC.EFTC(X).Q)” abstracts the intermediate functionality provided by the “External Formwork” of the sliding formwork system that has been moved beneath the deck segment to support the transitions of the states “Bottom & Side Rebar” and “Bottom Concrete” of the deck segment in “Cycle(X)”. The second set is the state interaction

"Contain(DSC.BC.Q, SFC.IFTC(X).A)". It abstracts the temporary support functionality provided by the bottom slab of the deck segment that has been cast (i.e. "DSC.BC.Q") so that the internal formwork can be slid into its position (i.e. "SFC.IFTC(X).A").

The third set of state interactions comprises "ContainedBy(DSC.TR, SFC.IFTC.Q)", "ContainedBy(DSC.T&SC, SFC.IFTC.Q)", and "ContainedBy (DSC.SD.A, SFC.IFTC.Q)". The former two interactions depict the intermediate functionality provided by the internal formwork to support the "Top Rebar" and "Side & Top Concrete" states. The third interaction implies that the temporary support functionality for the deck segment is still required while its strength is being developed to the degree that the deck segment can maintain its shape (i.e. "DSC.SD.A").

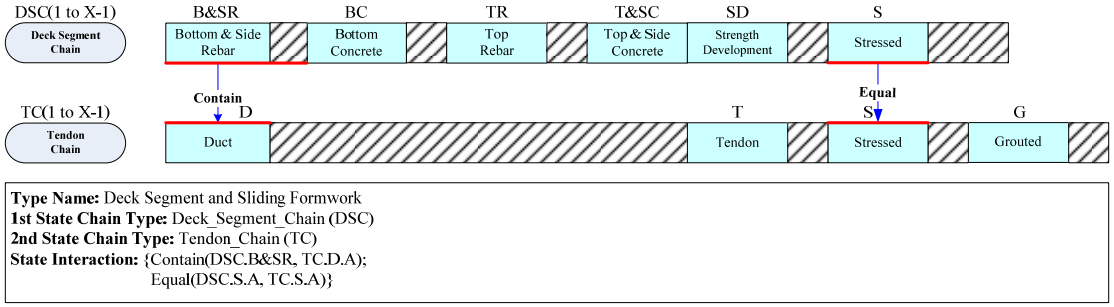



**Figure 7.18 State Interaction Type between Deck Segment and Tendon**

The functional interdependencies between the "Deck Segment Chain" and the "Tendon Chain" are shown in Figure 7.18. The "Tendon Chain" interrelates with the "Deck Segment Chain" through two interdependencies "Contain(DSC.B&SR, TC.D.A)" and "Equal(DSC.S.A, TC.S.A)". The first interdependency requires that the tendon duct installation be concurrently executed with the bottom and side rebar assembly of the deck segment, while the second interdependency indicates that the deck segments become stressed whenever the tendon clusters are being stressed. The first interdependency also implies that the installation of tendon duct (active phase "TC.D.A") can be finished later

than the installation of bottom and side rebar (active phase “DSC.B&SR.A”), but the former cannot start earlier than the latter since the former requires the support of in-progress rebar cage.

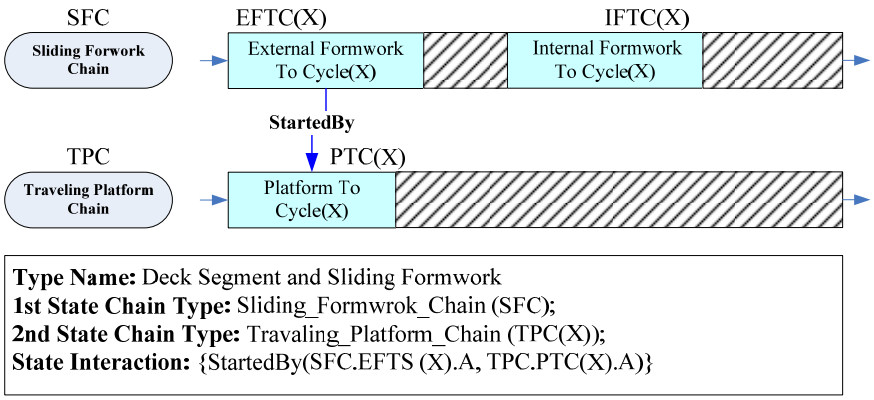



**Figure 7.19 State Interaction Type between Sliding Formwork and Traveling Platform**

The state interaction between the “Traveling Platform Chain” and the “Sliding Formwork Chain” is shown in Figure 7.19. The first interaction “StartedBy(SFC.EFTS (X).A, TPC.PTC(X).A)” indicates that the traveling platform and the sliding external formwork are simultaneously moved because the external formwork is bolted to the traveling platform. After they have been moved into place, the external formwork requires additional time to be adjusted to the right level and shape to accommodate the new segment to be constructed.

During the construction of the in-progress cantilever structure, an important intermediate functionality is to maintain its temporary support for the balanced cantilevers. Figure 7.20 shows the decomposition of the function provider system that provides the temporary support functionality in “Cycle(3)”, as an example, in which segments “SegL05” and “SegL06” will be cast and stressed (see Figure 7.3).

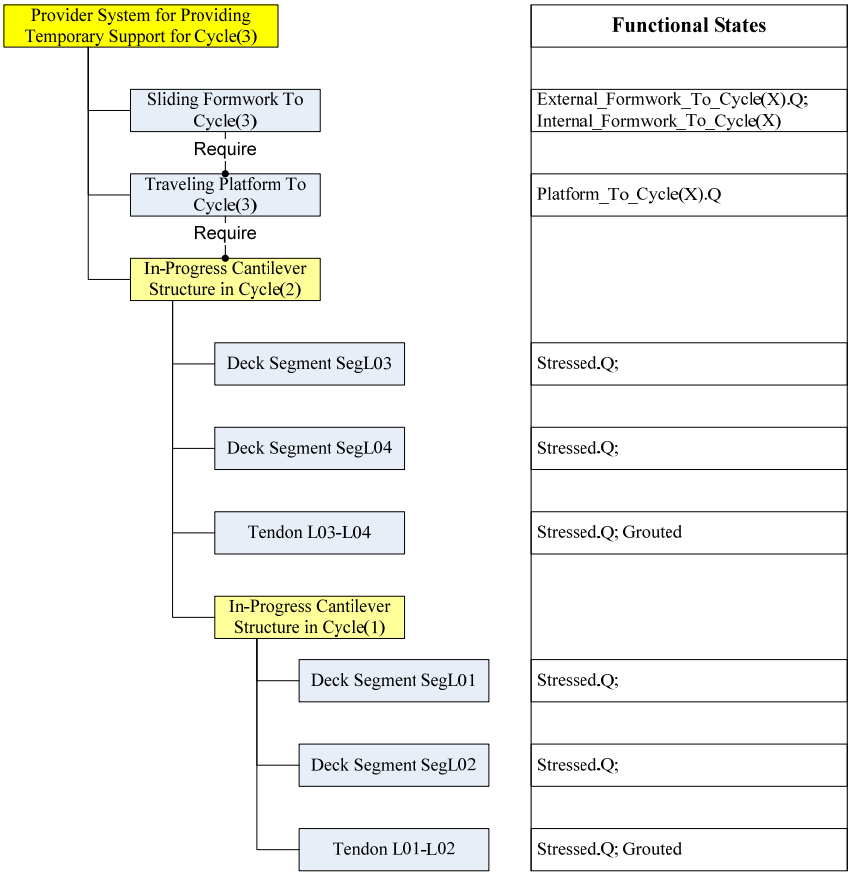



**Figure 7.20 Decomposition of Provider System and Functional States in Cycle(3)**

The direct support for a deck segment comes from the “Sliding Formwork”, which requires the support from “Traveling Platform”. The platform further requires the support from the embedded cantilever subsystem “In-Progress Cantilever Structure in Cycle(2)”. Meanwhile, the embedded subsystem is further decomposed into two deck segments and one tendon component as well as the earlier smaller embedded subsystem “In-Progress Cantilever Structure in Cycle(1)”. The functional states for each constituent component are presented in Figure 7.20.

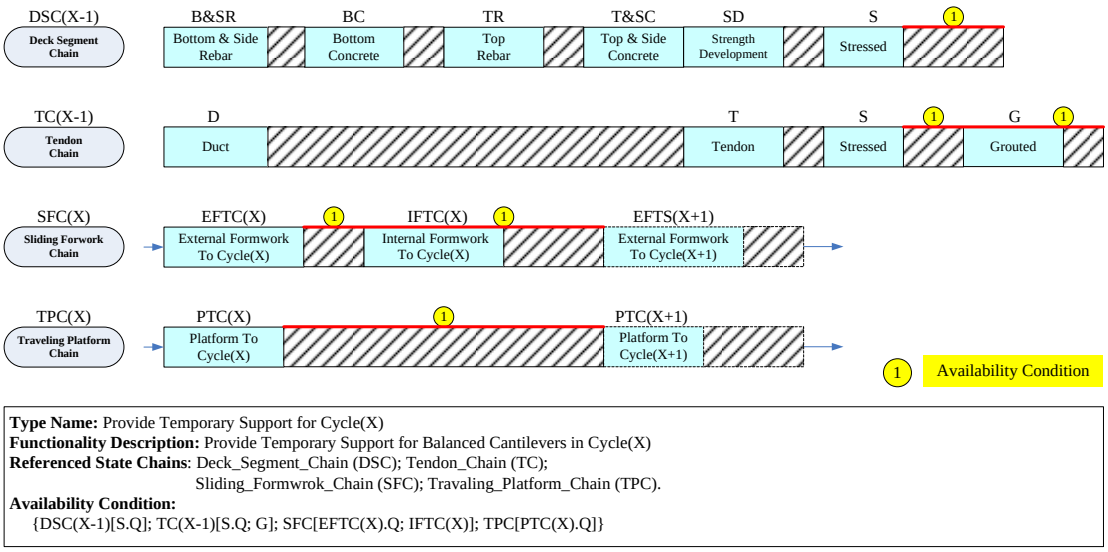



**Figure 7.21 Availability Type of Temporary Support Functionality in Cycle(X)**

Figure 7.21 shows that the functional states comprised in four state chains are organized into an availability condition (labeled by the circled number 1) to provide the temporary support functionality for constructing the deck segments in "Cycle(X)". The functional state of "Deck Segment (X-1)" is the "Stressed.Q" state, while the two functional states of "Tendon (X-1)" component are "Stressed.Q" and "Grouted" states. These functional states imply that the in-progress cantilever structure constructed in the "Cycle (X-1)" has been duly stressed and thus are able to support its balanced cantilever. This availability condition also sufficiently implies that the deck segments in the earlier cycles have been stressed as depicted in the embedded subsystems in Figure 7.20.

Based on this basic support functionality of the cantilever subsystem, two temporary facility components, namely the sliding formwork and the traveling platform can be advanced to provide the construction platforms for constructing the deck segments in "Cycle(X)". The traveling platform component has only one functional state "PTC(X).Q", indicating that it has completed its advancement in Cycle(X). Meanwhile, the functional states of the sliding formwork system are "EFTC(X).Q" and "IFTC(X)",

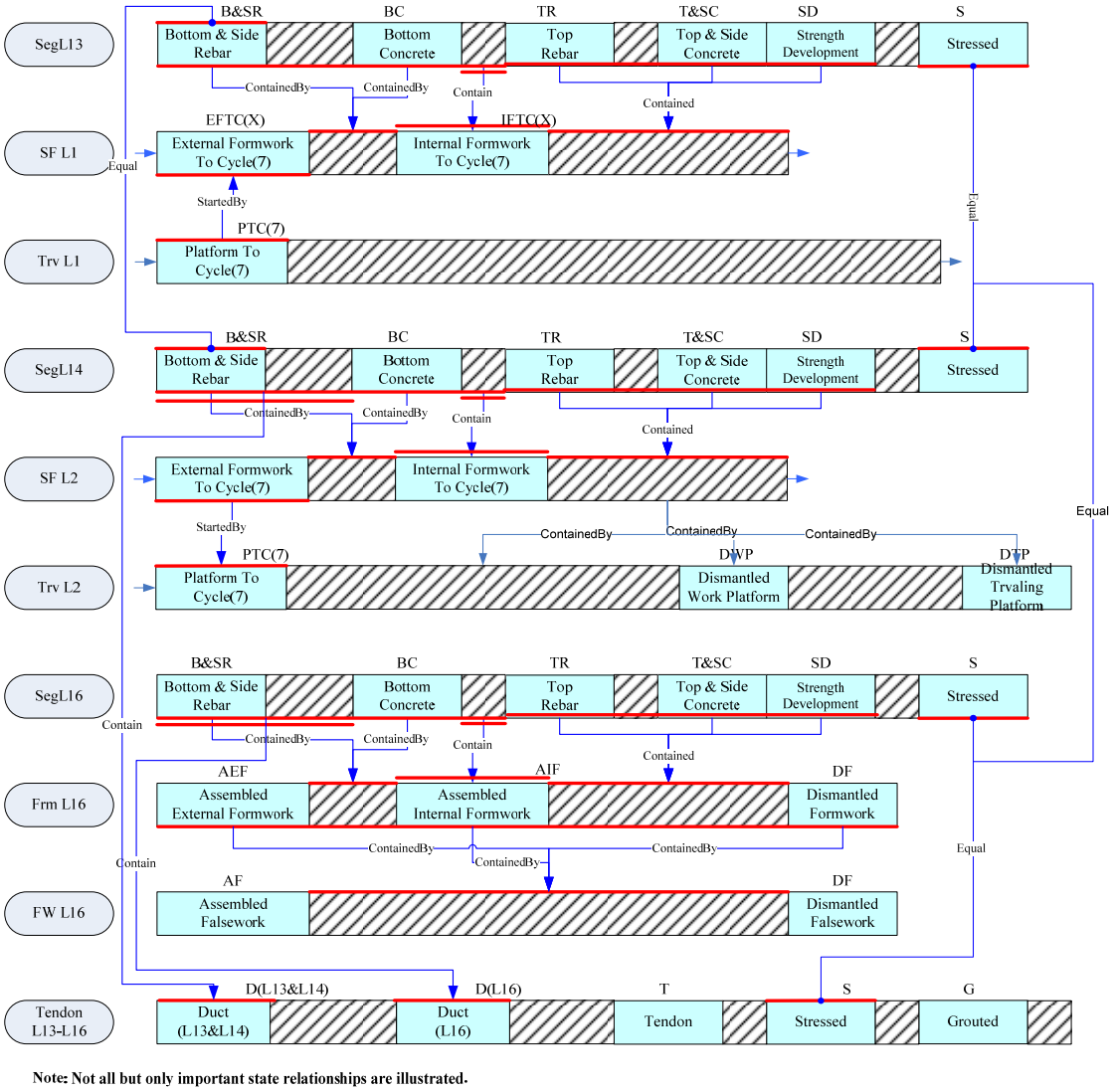
implying that the formwork parts have been advanced into their locations for Cycle(X) (See Figure 7.21). In this way, the traveling platform and the sliding formwork system, as well as the cantilever structure completed in “Cycle(X-1)”, work together to provide the temporary support functionality for constructing the deck segments in “Cycle (X)”.

This availability condition captures and also explains the temporal logics residing in the balance cantilever approach, specifying that the two cantilevers on each side of the pier can only be constructed outward with not more than one segment out of balance at any one time. Functionally, it depicts that the cantilever support can be maintained by the in-progress structure itself only after the tendon clusters have been stressed in Cycle(X-1), and then the traveling platform can be advanced to the subsequent location for Cycle(X).

### 7.1.4 Development of Component State Network Related to Cycle(7)

Using the intermediate function requirement knowledge elaborated in the preceding sections, the in-progress product core model (IPPCM) of the bridge deck can be developed for representing its in-progress configuration. Figure 7.22 shows a part of the IPPCM that is related to the Cycle(7) construction. The Figure shows four permanent components and six temporary components. These permanent components are three deck segments (“SegL13”, “SegL14”, and “SegL16”) and one tendon component (“TendonL13-L16”), while the temporary components are two sliding formworks (“SF L1” and “SF L2”), two traveling platforms (“Trv L1” and “Trv L2”), one formwork (“Frm L16”), and one falsework (“FW L16”). Meanwhile, Figure 7.22 also illustrates the pertinent state relationships between these product components with respect to the construction life cycle of each product component, the functional interdependencies

between two in-progress components, and the concurrent relationships between the component states contained in the same work package.




Note: Not all but only important state relationships are illustrated.

**Figure 7.22 In-Progress Product Core Model for Cycle(7)**

The development of the construction life cycles of the product components in Figure 7.22 can be facilitated by using the state chain knowledge addressed in the preceding section. For example, the construction life cycles of three deck segments can be derived from the “Deck Segment Chain” type illustrated in Figure 7.12. Similarly, the construction life cycle of “Tendon L13-L16” can be developed from the “Tendon Chain”

type shown in Figure 7.14. The construction life cycles of two sliding formworks and two traveling platforms can also be initialized from “Sliding Formwork Chain” (Figure 7.15) and “Traveling Platform Chain” type (Figure 7.16), respectively. Additionally, the construction life cycles of “Frm L16” and “FW L16” can likewise developed.

Furthermore, some construction life cycles are further developed for accommodating the specific construction sequence. Specifically, the “Duct” state in the construction life cycle of “Tendon L13-L16” is split into two state “Duct(L13&L14)” and “Duct(L16)” since the duct are assembled in two steps associated with two construction activities “Install Bottom and Side Rebar L13&L14” and “Install Bottom and Side Rebar L16”, respectively (See Figure 7.23), between which there is a gap. The construction life cycle of “Trv L2” is also added a “Dismantled Work Platform” state following its “Platform To Cycle(7)” state to describe that the work platform of “Trv L2” should be dismantled after the construction of “SegL14” in order to release the space for erecting “FW L16”.

The functional dependencies between the in-porgress product components illustrated in Figure 7.22 can be represented in terms of cross-component state relationships, especially “Contain” and “ContainedBy” relationships. The establishment of these state relationships can be facilitated by the state interaction knowledge like that explained in Figures 7.17 to 7.19. For example, the state relationship “ContainedBy (SegL14.T&SC.A, SF_L2.IFTC(7).Q)” can be derived using the knowledge explained in Figure 7.17, indicating that casting concrete of the top and side slabs of “SegL14” requires the temporary support from the internal formwork of the slide formwork “SF L2” that has been slid to the designed location for “SegL14”.

Figure 7.22 also illustrates the concurrent transition of a number of component states contained in the same work package in terms of “Equal” relationships between their active phases. For example, the active phases “Stressed.A” of deck segments “SegL13” “SegL14” and “SegL16” and tendon “Tendon L13-L16” are contained in the work package of construction activity “Stress Deck L13-L16” so there is an “Equal” relationship between each two of these four active phases (Figure 7.22).

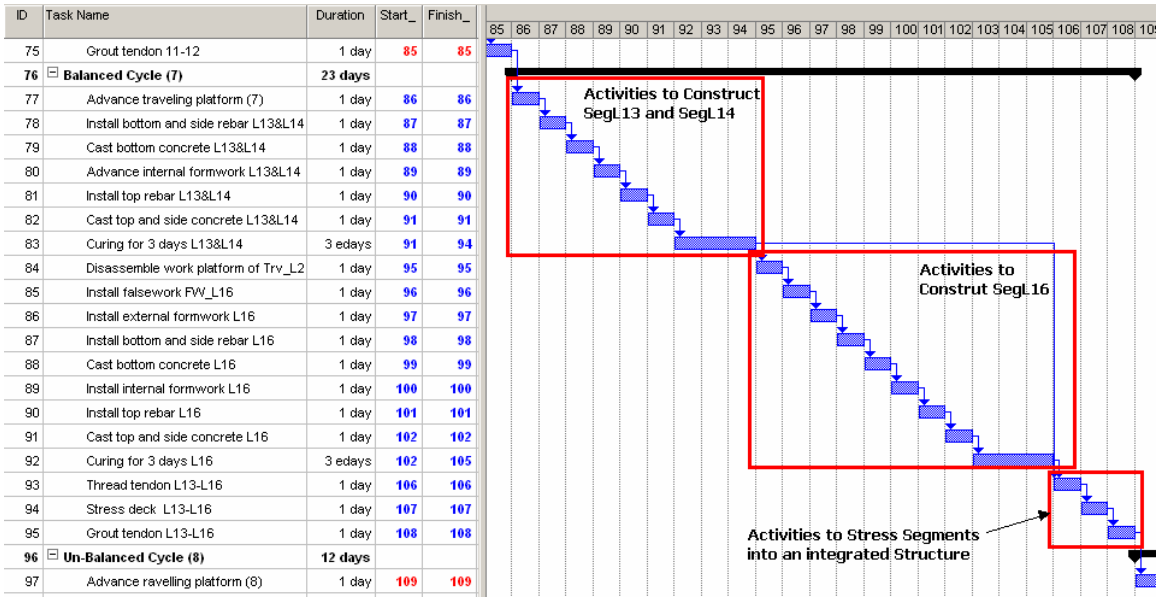



**Figure 7.23 Original CPM Schedule for Cycle(7)**

By using the Product Oriented Scheduling Technology (POST) introduced in Section 3.4, the temporal attributes of the component states in the component state network illustrated in Figure 7.22 can be derived from the corresponding CPM schedule for Cycle(7) as shown in Figure 7.23. The construction activities in Figure 7.23 are grouped into three blocks: constructing “SegL13” and “SegL14” (ID 77 to ID 83), constructing “SegL16” (ID 84 to ID 92), and post-tensioning the segments into an integrated structure (ID 93 to ID 95). Additionally, the natural curing processes and the

logistic activities for advancing the traveling platform and the sliding formworks are also listed in the CPM schedule.

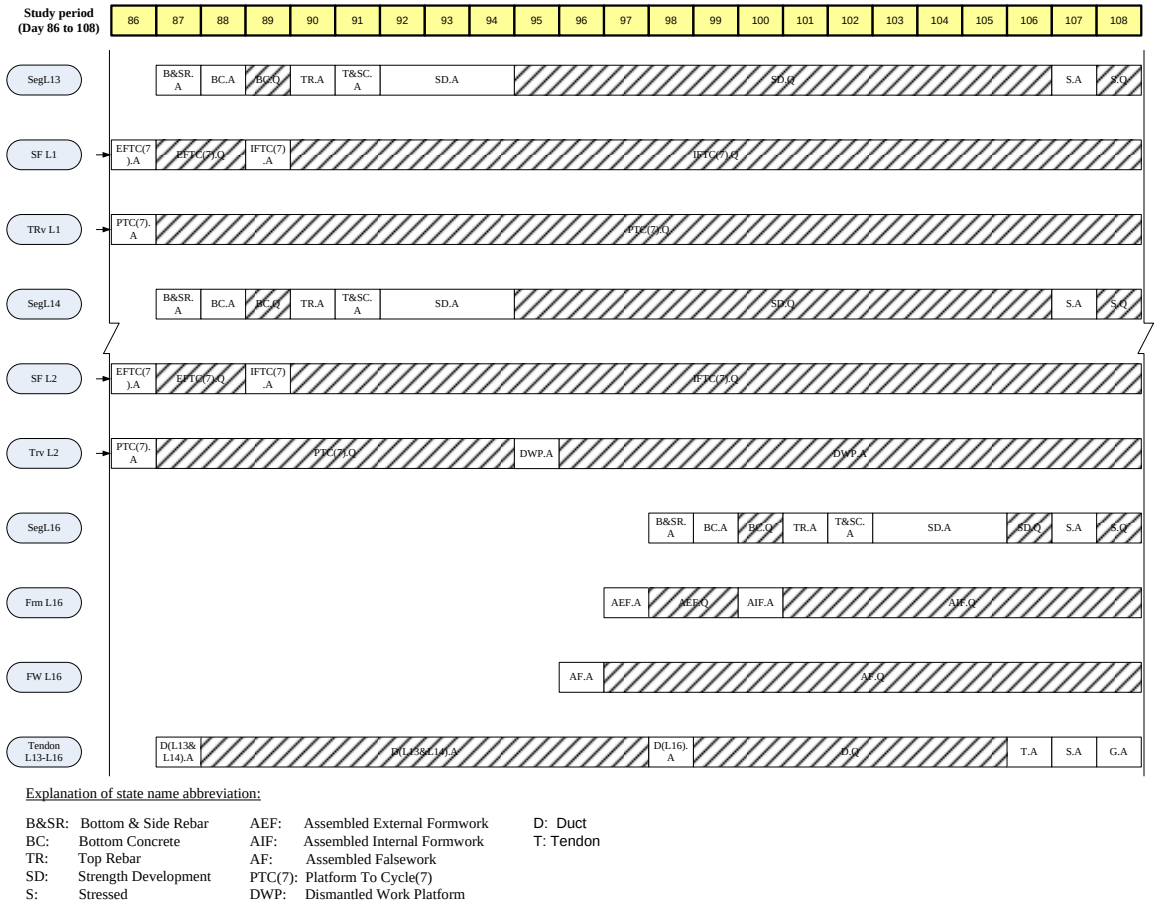
**Table 7.1 Work Packages of Activities for Cycle(7)**

| ID | Activity | Start (D) | Finish (D) | Work Package |
|---|---|---|---|---|
| 77 | Advance traveling platform (7) | 86 | 86 | Trv_L1.PTC(7).A, Trv_L2.PTC(7), SF_L1.EFTC(7).A, SF_L2.EFTC(7).A |
| 78 | Install bottom and side rebar L13&L14 | 87 | 87 | SegL13.B&SR.A, SegL14.B&SR.A., TendonL13-l16.D(L13&L14).A |
| 79 | Cast bottom concrete L13&L14 | 88 | 88 | SegL13.BC.A, SegL14.BC.A |
| 80 | Advance internal formwork L13&L14 | 89 | 89 | SFL1.IFTC(7).A, SFL2.IFTC(7).A |
| 81 | Install top rebar L13&L14 | 90 | 90 | SegL13.TB.A, SegL14.TB.A, |
| 82 | Cast top and side concrete L13&L14 | 91 | 91 | SegL13.T&SC.A, SegL14.T&SC.A |
| 83 | Curing for 3 days L13&L14 | 91 | 94 | SegL13.SD.A, SegL14.SD.A |
| 84 | Disassemble work platform | 95 | 95 | Trv_L2.DWP.A |
| 85 | Install falsework L16 | 96 | 96 | FW_L16.AF.A |
| 86 | Install external formwork L16 | 97 | 97 | Frm_L16.AEF.A |
| 87 | Install bottom and side rebar L16 | 98 | 98 | SegL16.B&SR.A, TendonL13-L16.(DL16).A |
| 88 | Cast bottom concrete L16 | 99 | 99 | SegL16.BC.A |
| 89 | Install internal formwork L16 | 100 | 100 | Frm_L16.AIF.A |
| 90 | Install top rebar L16 | 101 | 101 | SegL16.TR.A |
| 91 | Cast top and side concrete L16 | 102 | 102 | SegL16.T&SC.A |
| 92 | Curing for 3 days 16 | 102 | 105 | SegL16.SD.A |
| 93 | Thread Tendon L13-L16 | 106 | 106 | TendonL13-16.T.A |
| 94 | Stress deck L13-L16 | 107 | 107 | SegL13.S.A, SegL14.S.A, SegL16.S.A, TendonL13-L16.S.A |
| 95 | Grout tendon L13-L16 | 108 | 108 | TendonL13-L16.G.A |

**Note: The component states in the work packages are abbreviated by the initials of each word. For example, the "Bottom Concrete" state is abbreviated by "BC". The abbreviations of the states in the work package column can reference Figure 7.22.**

The construction activities in the CPM schedule will map their temporal attributes onto the corresponding component states through the corresponding work packages as shown in Table 7.1. The Table presents the start, finish, and work packages of each activity comprised in the CPM schedule for Cycle(7). The component states associated with the right cantilever structure is intentionally omitted for briefing the statements. For example, the work package of the activity "Install Bottom and Side Rebar L13&L14" (ID 78) contains altogether three active state phases, namely the active phases of the "Bottom

and Side Rebar" states of both "SegL13" and "SegL14" and the active phase of the "Duct (L13 and L14)" state of "TendonL13-L16". The other work packages can be likewise defined. Using POST, the temporal attributes of each associated component state can be derived from the start and finish times of the associated activities. Figure 7.24 illustrates the derivation results in the format of a bar chart within a period from the start of Day 86 to the end of Day 108. These derived temporal attributes will be used for intermediate function analysis in the following sections.



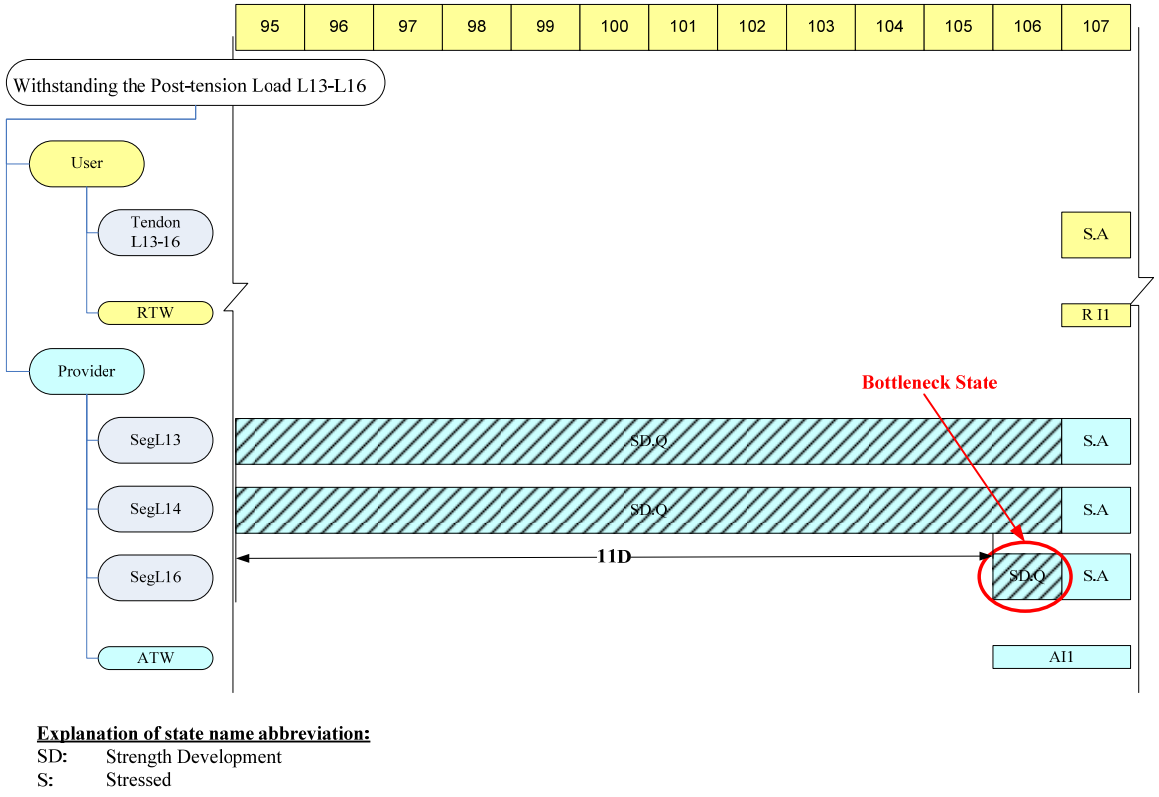

Explanation of state name abbreviation:

| | | | | | |
|---|---|---|---|---|---|
| B&SR: | Bottom & Side Rebar | AEF: | Assembled External Formwork | D: | Duct |
| BC: | Bottom Concrete | AIF: | Assembled Internal Formwork | T: | Tendon |
| TR: | Top Rebar | AF: | Assembled Falsework | | |
| SD: | Strength Development | PTC(7): | Platform To Cycle(7) | | |
| S: | Stressed | DWP: | Dismantled Work Platform | | |

**Figure 7.24 In-Progess Product Core Model for Cycle(7)**

### 7.1.5 Identification of Bottleneck State in Cycle(7)

Due to delays in upstream activities, the contractor has to expedite the construction schedule. Analysis of the schedule suggests opportunities to expedite the seventh construction cycle to complete the bridge deck in time. After examining the intermediate functionalities associated with each critical activity in Cycle(7), the planning engineers find that the commencement of the critical activity “Stress Deck L13-L16” is constrained by the intermediate functionality to withstand the post-tension load.




**Explanation of state name abbreviation:**
SD: Strength Development
S: Stressed

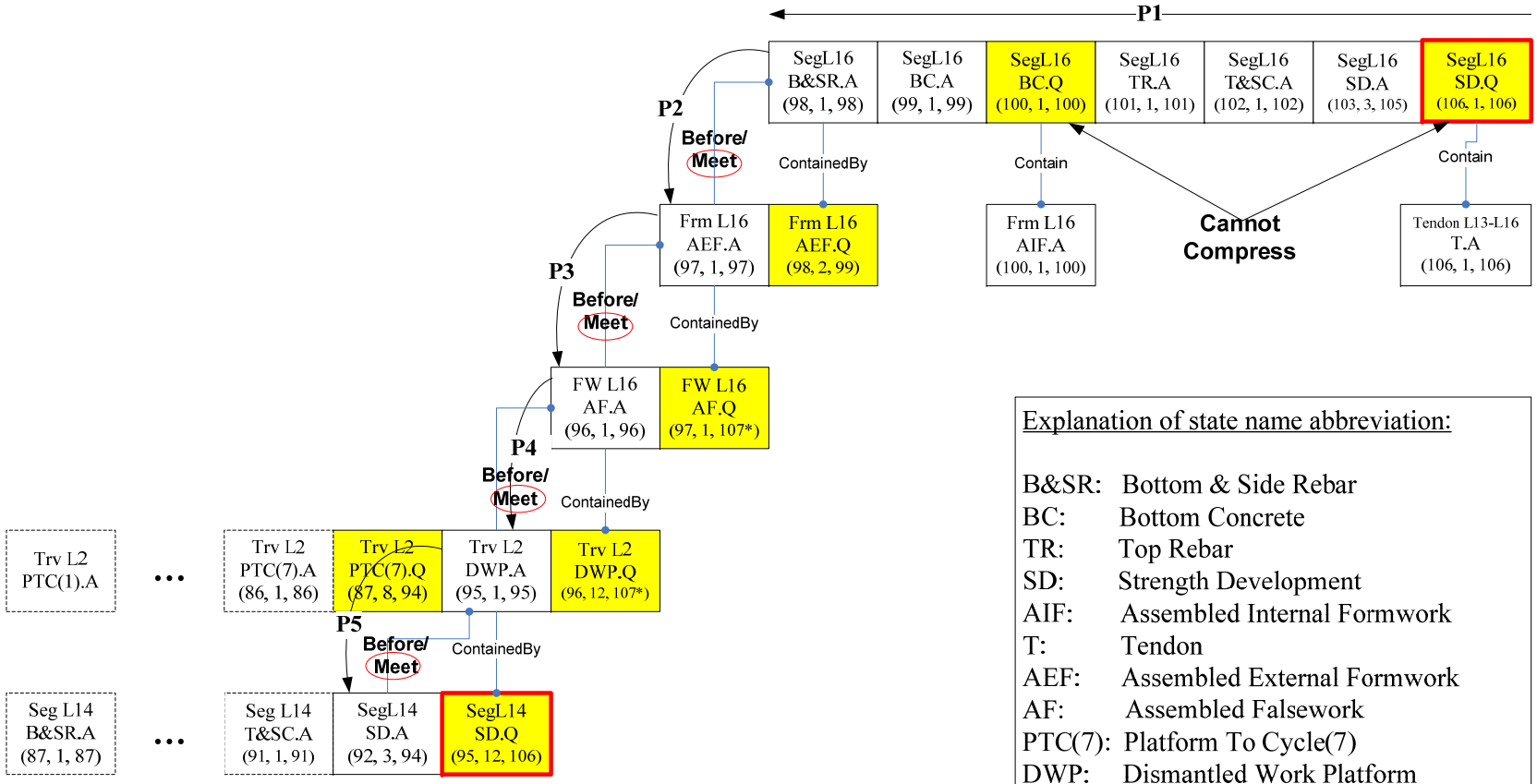
**Figure 7.25 Intermediate Function for Post-tensioning Tendon L13-L16**

Figure 7.25 shows the composition of the intermediate function for post-tensioning “Tendon L13-L16”. The function user is the “Tendon L13-L16” component, which requires the associated intermediate functionality to withstand the post-tension load when it is being stressed (“Stressing.A”). The function provider comprises three

deck segments “SegL13”, “SegL14”, and “SegL16”. The intermediate function is available when the three deck segments are in their quiescent phase of the “Strength development” states (“SD.Q”) and active phases of the “Stressed” states (“S.A”). The “S.A” should be included in the functional state, since when the tendon component is in its “S.A”, the corresponding deck segment must be in their “S.A”.

The requirement time-window and the availability time-window can be derived using the Equations (5.1) and (5.2). Figure 7.25 shows that the requirement time window is fully contained by the corresponding availability time-window, indicating that when post-tensioning “Tendon L13-L16”, the cantilever structure from SegL13 to SegL16 is available to withstand the post-tension load. Furthermore, Figure 7.25 also shows that “SD.Q” of “SegL16” is 11 days later than the “SD.Q” phases of both “SegL13” and “SegL14”, indicating that “SD” of “SegL16” is a bottleneck state that delays the availability of the required intermediate functionality. If this bottleneck state can be advanced, the intermediate functionality for withstanding post-tension load can be realized earlier, leading to the earlier commencement of the critical activity “Stress Deck L13-L16”. Thus, the construction of Cycle(7) can be expedited.

### 7.1.6 Analysis of Bottleneck State




*** Assuming that the end of the study period is the end of Day 107**

**Figure 7.26 Precedence Chain for Bottleneck State**

In order to explore the feasibility of advancing the bottleneck state, the component states/phases that constrain the start of the bottleneck state should be explored in terms of a chain of precedence path. The component state/phases along a precedence chain are associated by either a “Meet” or “Before” relationship. This precedence chain can be used to locate opportunities for advancing the bottleneck state.

Figure 7.26 shows the precedence chain linking the “SD” states of “SegL14” and “SegL16”. The precedence path can be first traced from the bottleneck state “Strength Development” of “SegL16” upstream along its construction life cycle until its initial active phase “Bottom & Side Rebar.A” (“B&SR.A”), which is the labeled by the precedence path “P1” in Figure 7.26. There are altogether five active state phases and two quiescent phases along “P1”. These seven state phases constrain the advancement of the bottleneck state “SD”. If either the preceding quiescent phases can be compressed or the preceding active phases can brought forward, it is possible to advance the bottleneck state. Unfortunately, the two quiescent phases along the precedence path “P1”, namely “BC.Q” (quiescent phase of state “Bottom Concrete”) and “SD.Q” (quiescent phase of state “Strength Development”), cannot be compressed since the duration interval of the former is totally used for installing the internal formwork (active phase “AIF.A”) of “Frm L16” and the duration interval of the latter is fully required for threading “Tendon L13-L16” (active phase “T.A”). These two functional interdependencies are represented by two “ContainedBy” state relationships in Figure 7.26.

The remaining opportunity for advancing the state chain along the precedence path “P1” should be explored by locating the associated component states in other construction life cycles that constrain the commencement of active phase “B&SR” of

“SegL16”, since it is the initial state in the construction life cycle of “SegL16”. Figure 7.26 shows that “B&SR.A” of “SegL16” is “Contained By” the quiescent phase of state “Assembled External Formwork” (“AEF.Q”) of formwork “Frm L16” since the installation of side and bottom rebar of “SegL16” requires the temporary support from the assembled external formwork of “Frm L16”. Meanwhile, the corresponding active phase “AEF.A” “Meets” the quiescent phase “AEF.Q” along the construction life cycle of “Frm L16”. Thus, it can be inferred that the active phase “AEF.A” of “Frm L16” should either be “Before” or “Meet” the quiescent phase “B&SR.A” of “SegL16”, i.e. “Before/Meet(Frm_L16.AEF.A, SegL16.B&SR)”, implying that the start of bottom and side rebar installation (“B&SR.A”) of “SegL16” is restricted by the installation of the external formwork installation (“AEF.A”) of “Frm L16”. Furthermore, their temporal attributes indicate that the former “Meets” the latter. Therefore, the advancement of “B&SR.A” of “SegL16” depends on whether “AEF.A” of “FrmL16” can be brought forward. This precedence path from “SegL16.B&SRS.A” to “Frm_L16.AEF.A” is labeled as “P2” in Figure 7.26.

Similarly, since “AEF.A” is the initial active phase in the construction life cycle of “Frm L16” (see Figure 7.23), its advancement can be constrained by the associated states of other components. Specifically, “AEF.A” of “Frm L16” is contained by “AF.Q” of “FW L16”. Again, a “Before/Meet” relationship “Before/Meet(FW_L16.AF.A, Frm_L16.AEF.A)” can be inferred, labeled by the precedent path “P3” in Figure 7.26. The temporal attributes of the two associated state phases indicate that “FW_L16.AF.A” meets “Frm_L16.AEF.A”, meaning that “AEF.A” of “Frm L16” can be advanced only if “AF.A” of “FW L16” can be brought forward.

Likewise, the precedence path “P4” can be derived from two existing state relationships: “ContainedBy(FW_L16.AF.A, Trv_L2.DWP.Q)” and Meet(Trv_L2.DWP.A, Trv_L2.DWP.Q) as depicted in Figure 7.26. The resultant state relationship “Before/Meet(Trv_L2.DWP.A, FW_L16.AF.A)” indicates that the assembling (“AF.A”) of the falsework “FW L16” is restricted by the dismantling of the work platform (“DWP.A”) of the traveling platform “TrvL2”, since the latter can release the space for the former in order to prevent the potential space collision as explained in Figure 7.11. The temporal attributes of these two active phases show that “Trv_L2.DWP.A” meets “FW_L16.AF.A”, indicating that the advancement of “AF.A” of “FW L16” depends on whether “DWP.A” of “Trv L2” can be brought forward.

The inference of the precedent path “P5” from the active phase “DWP.A” of “Trv L2” to the active phase “SD.A” of “SegL14” is similar to the inference of precedence path “P4” (Figure 7.26). The work platform of “Trv L2” can be dismantled after “SegL14” has achieved its 3-day strength, as represented by the “ContainedBy” relationship “ContainedBy(Trv_L2.DWP.A, SegL14.SD.Q)”. This “ContainedBy” relationship and the “Meet” relationship “Meet(SegL14.SD.A, SegL14.SD.Q)” give rise to the “Before/Meet” relationship “Before/Meet(SegL14.SD.A, Trv_L2.DWP.A)”. The temporal attributes again indicates that “SegL14.SD.A” “Meets” “Trv_L2.DWP.A” so that the dismantling of the work platform of “Trv L2” can only be advanced if the 3-day strength development (hence the casting) of “SegL14” can be brought forward.

Consequently, the preceding states that restrict the start of the bottleneck state are identified as a chain of precedent paths from “P1” to “P5”. At the same time, the associate relationship between the bottleneck state “SD” of “SegL16” and state “SD” of

“SegL14” can also be inferred from the precedent chain, indicating that “SD.Q” of “SegL16” cannot be earlier than “SD.A” of “SegL14”. Furthermore, from the temporal attributes of the state phases along the precedent chain “P1” to “P5”, a gap of 11 days can be obtained. This is equal to the difference between the start points of the duration intervals of the “SD.Q” phases of “SegL16” and “SegL14”, implying that the gap between these two quiescent phases cannot be compressed.

### 7.1.7 Alternative Construction Method for Advancing Bottleneck State

Further analysis of the IPPCM shows that the “SD.A” of “SegL14” cannot be advanced because of the upstream construction life cycles. The remaining option is to explore an alternative construction method to synchronize the construction of “SegL14” and “SegL16”. Through the 4D simulation frame shown in Figure 7.11, the height clearance for deck segments “SegL14” and “SegL16” was determined to be only 2.5m to 3.5m from the bank level. This height makes it feasible to erect of a longer falsework (“FW L14-L16”) for supporting the concurrent construction of both “SegL14” and “SegL16”. In this way, the construction life cycles of “SegL13”, “SegL14” and “SegL16” can be synchronized in order to advance the realization of the intermediate functionality for withstanding the post-tension load exerted by “Tendon L13-L16”.

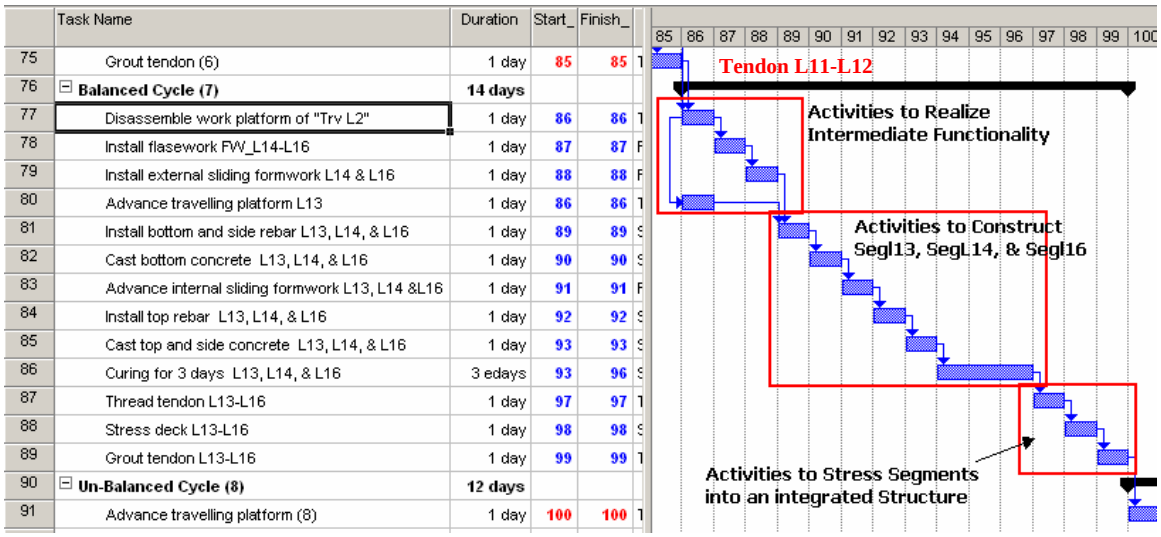



**Figure 7.27 Shortened CPM Schedule for Construction Cycle(7)**

Figure 7.27 shows the revised construction schedule for Cycle(7), and Figure 7.28 depicts the corresponding component state network. The modified construction sequence indicates that after the grouting (active phase “G.A”) of “Tendon L11-L12” is completed, the work platform of “Trv L2” can be dismantled to clear the space for assembling the falsework “FW L14-L16” (“AF.A”). The external formworks of sliding formwork “SF L2” and the original formwork “Frm L16” can be put in place with the support from “FW L14-L16” for concurrently constructing “SegL14” and “SegL16”.

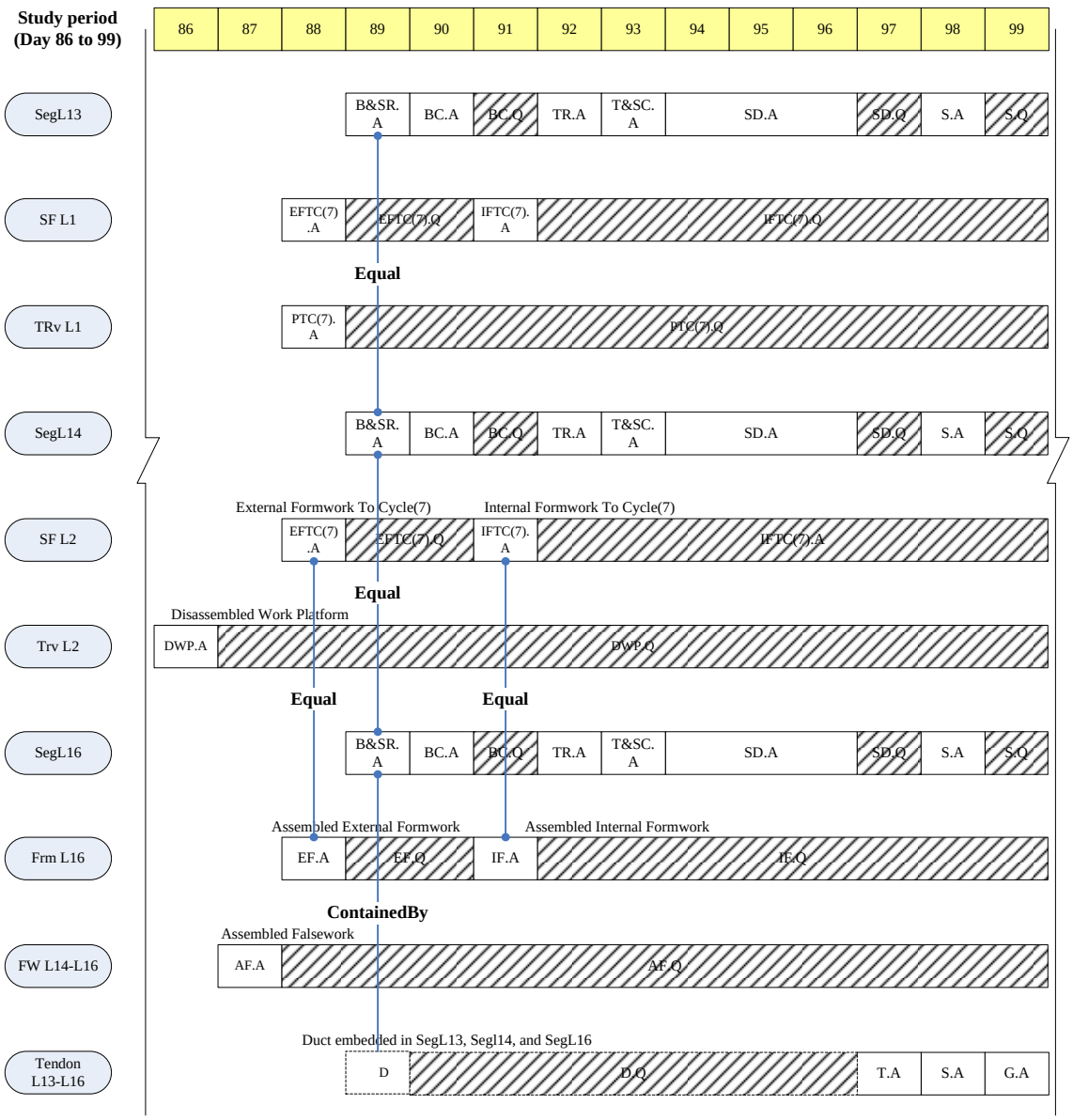



**Figure 7.28 In-Progress Product Model for Cycle (7) of Improved Schedule**

Two pairs of active phases, namely “EFTC(7).A” of “SF L2” with “AEF.A” of “Frm L16” and “IFTC(7).A” of “SF L2” with “AIF.A” of “Frm L16”, can be synchronized through the use of two “Equal” relationships, indicating that the external and internal formworks of “SF L2” and “Frm L16” are installed together in the same work package. Additionally, the duct installation of “TendonL13-L16”,which was split into two states “Duct(L13&L14)” and “Duct(L16)” in the original schedule (see Figure

7.24), is now completed in one step in the revised schedule contained by the active phases of “B&SR” of “SegL13”, “SegL14” and “SegL16”, which have been synchronized in the same work package of “Install Bottom and Side Rebar L13, L14, & L16” in Figure 7.27. Consequently, the intermediate functionality for post-tensioning “Tendon L13-L16” can be greatly advanced, reducing construction period by 9 days, which is more than an eight percent savings over the time required for constructing the balanced cantilever structure.

## 7.2 Case Study Two: Construction of Entrance Gate of Nursing Home

### 7.2.1 Original Construction Schedule

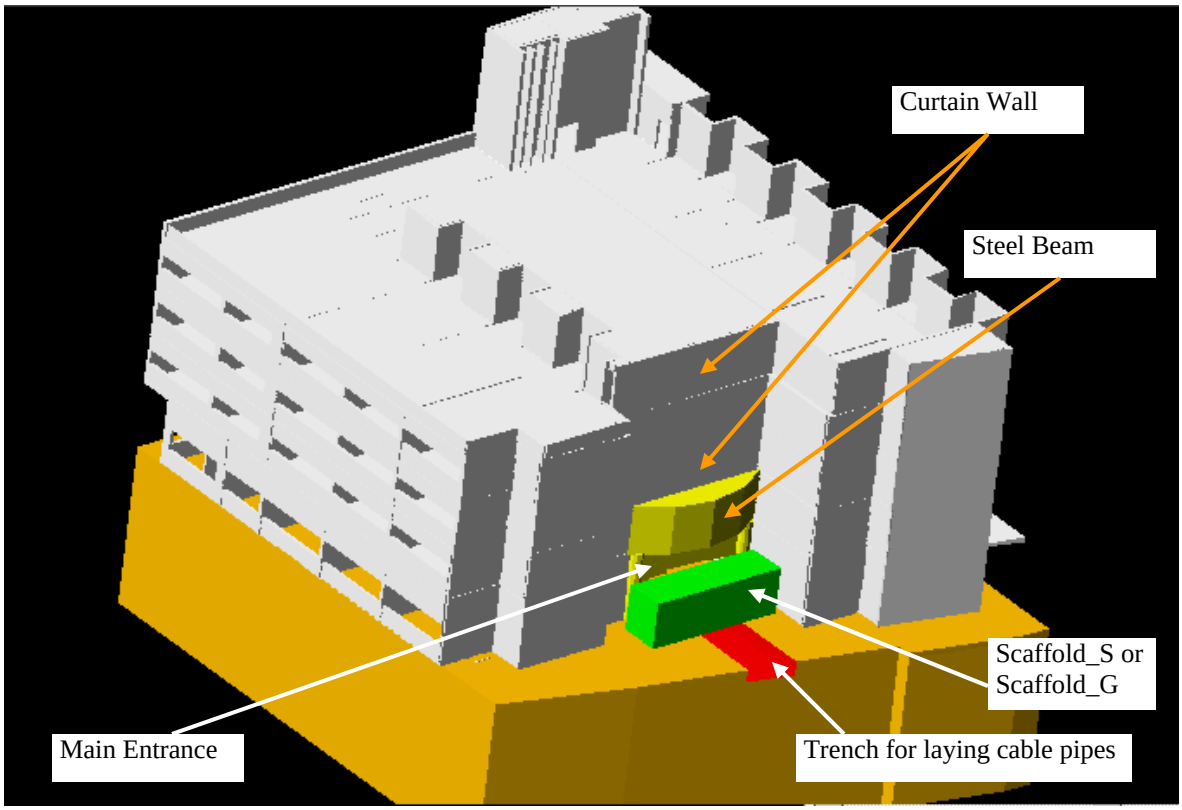



**Figure 7.29 3D model of Nursing House Showing Main Entrance**

This case study presents the schedule to construct the main entrance of a nursing home. One important task is to design and build the glass works for the front facade, which is closely related to the construction of the curved steel beam above the main entrance. The scope of the glass works includes the glass door of the main entrance and the curtain walls of the upper floors (see Figure 7.29). The steel beam is fabricated in segments off-site and assembled on-site by welding using a scaffold as temporary support.

The design and construction of the glass works are subcontracted to the subcontractor “SubCon_1”, while the steel beam works are subcontracted to another subcontractor “SubCon_2”. The cable pipe laying is performed by the main contractor

“MainCon”. The trench work for cable pipe laying is shown in Figure 7.29. “SubCon_1” will require the as-built information of the steel beam for designing the shop drawings for the front glass door and curtain wall glass panels. Additionally, the Finish-to-finish precedence relationship between “Site Survey” and “Design Shop Drawing” of the glass works has a lag time of five days (labeled in Figure 7.30) to allow for completion of the shop drawing design after acquiring the as-built information of the completed steel beam.

The master schedule in Figure 7.30 shows that the construction of the glass works cannot commence before the end of Day 240. Otherwise, the path space through the main entrance required by the MainCon’s works (from the start of Day 227 to the end of Day 240) will be blocked by “Scaffold_G”. Meanwhile, the glass subcontractor “SubCon_1” should finish site demobilization before the end Day 277 so that the downstream cladding works that should start on Day 278 will delayed (See Figure 7.30).

The original schedule in Figure 7.30 also shows that “SubCon_2” (Steel Beam subcontract) “Demobilizes” at the end of Day 224 to clear the path space near the main entrance in time for downstream works to be executed by “MainCon” who requires the main entrance for access from the start of Day 227 to the end of Day 240. During the demobilization, “SubCon_2” dismantles, on Days 223 to 224, “Scaffold_S” that was erected for designing, assembling and welding the steel beam segments. Another scaffold “Scaffold_G” was later erected on Day 241 for measuring as-built information and for executing glass works, and finally “SubCon_1” finishes their demobilization on Day 274.

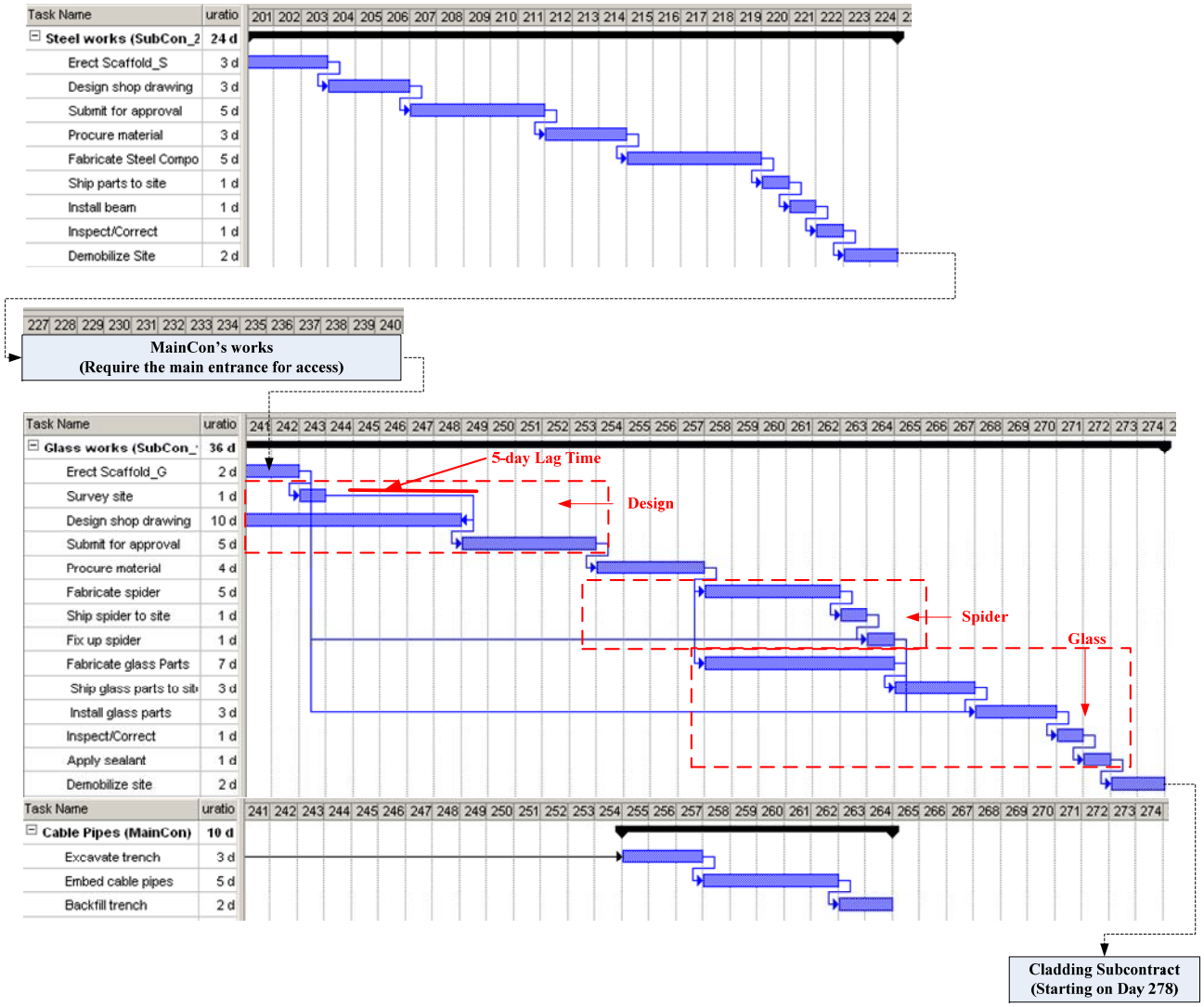



**Figure 7.30 Original Schedule for Beam, Glass Works, and Cable Pipes**

### 7.2.2 Intermediate Function Analysis for Temporary Support in Original Schedule

Figure 7.31 illustrates the function user “User 1” of the temporary support function “Funciton_1” comprising two labor components “Assistant Designer” and “Glass Workers”. The temporary support functionality is required by the “Assistant Designer” for measuring the as-built information of the completed steel beam and later required by the “Glass Workers” for constructing spider and glass components (fixing up spiders, assembling glass panels, checking and correcting assembled works, and applying sealant).

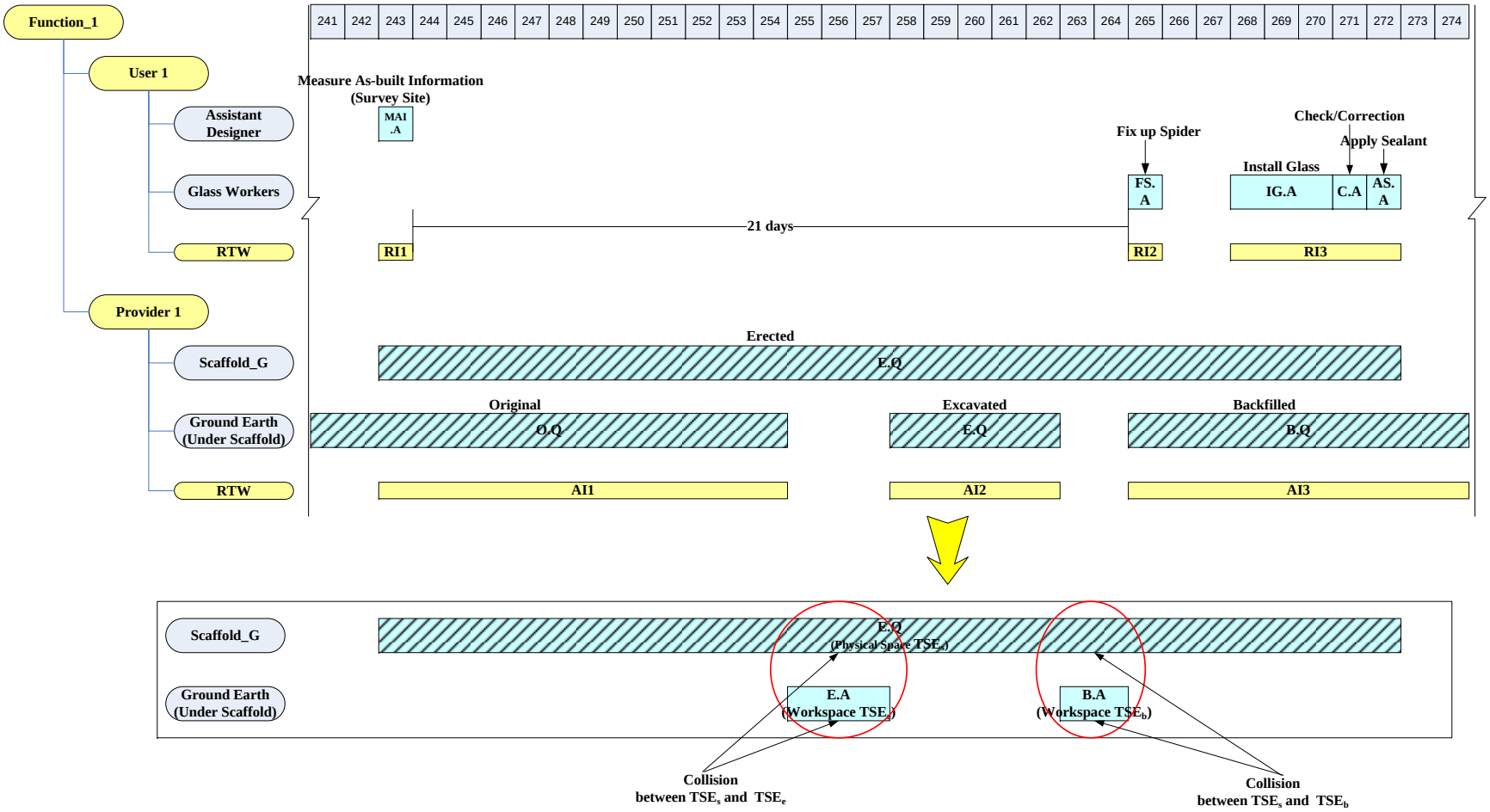



**Figure 7.31 Unavailable Temporary Support in Original Schedule**

Figure 7.31 also shows the function provider “Provider 1” of “Function_1” comprising “Scaffold_G” and “Ground Earth” (under “Scaffold_G”). The temporary support functionality be made available only when “Scaffold_G” is in its “Erected.Q” quiescent phase (after the erection and before the dismantling) and when “Trench for Pipe Cabling” is its three functional states: “Original.Q” (before excavation), “Excavated.Q” (after excavation and before backfill), and “Backfilled.Q” (after the backfill).

The temporal attributes of the requirement and functional states of “Function_1” are also shown in Figure 7.31 as indicated by the earlier schedule in Figure 7.30 via the corresponding work packages. Figure 7.31 shows that the requirement time “RTW” (containing intervals “RI1” to “RI3”) is totally covered by the corresponding availability time-window “ATW” (containing intervals “AI1” to “AI3”), where the requirement interval “RI1” is contained by “AI1” and “RI3” contained by “AI3”. Additionally, the functional state phase “Excavated.Q” of “Ground Earth” is not required by the user.

Figure 7.31 indicates that two active phases “Excavated.A” (“E.A”) and “Backfilled.A” (“B.A”) of “Ground Earth” between its three functional states are also contained by the long duration interval of functional state “Erected.Q” of “Scaffold_G”. The figure also shows that the workspace entities associated with these two active phases are denoted “$TSE_e$” and “$TSE_b$”, respectively, and their existence periods are equal to the active duration of the two associated active phases, i.e. I(255, 257) and I(263, 264). “$TSE_s$” denotes the physical space entity associated with quiescent phase “Erected.Q” of “Scaffold_G”, and its existence period of “$TSE_s$” is equal to the quiescent interval of

"Scaffold_G.Erected.Q", i.e. I(243, 272). The temporal topological relationships between these temporal space entities can be derived as follows:

**$TTR(TSE_e, TSE_s)$ = Temporal Intersection**

**$EP(TTR(TSE_e, TSE_s)) = EP(TSE_e) \cap EP(TSE_s) = I(255, 257) \cap I(243, 272)$**

**$= I(255, 257)$**

**$TTR(TSE_b, TSE_s)$ = Temporal Intersection**

**$EP(TTR(TSE_b, TSE_s)) = EP(TSE_b) \cap EP(TSE_s) = I(263, 264) \cap I(243, 272)$**

**$= I(263, 264)$**

The derived two temporal intersection relationships have non-null existence periods, indicating that the workspaces for excavating and backfilling the trench in "Ground Earth" would be blocked by the physical space occupied by the erected "Scaffold_G", thus rendering the construction schedule infeasible.

### 7.2.3 Scheduling Alternatives for Resolving Conflict

Several planning alternatives can be considered for resolving the aforementioned scheduling conflict. The first alternative is to postpone the measurement of the as-built information of steel beam until the trench for pipe cabling is backfilled so that "Scaffold_G" can be erected later and not collide with the excavation and the backfilling. However, this will result in a delay in the measurement of the as-built information and hence the shop drawing design and the final installation of the glass works.

The second alternative is to postpone the excavation works until "Scaffold_G" is dismantled, thus relieving the collision. However, this will delay the laying of the electrical cable works and affect the installation of the cladding works at the main

entrance which also require the work space in the vicinity of the trench. Consequently, the project will be delayed.

The third alternative suggests that “Scaffold_G” be erected earlier and then dismantled after the measurement of the as-built information before Day 227 to make way for the access of the “MainCon” through the main entrance. After the trench is backfilled, “Scaffold_G” will be re-erected. This alternative may not significantly delay the delivery of the various subcontract works but will cost more for the temporary facilities to “SubCon_1”. Such alternatives often happen on site if the conflict is not detected early in planning.

Figure 7.31 shows that there is a 21-day gap between the first requirement interval (for measuring the as-built information) and the second requirement interval (for fixing spiders) in the discontinuous requirement time-window. This indicates that “Scaffold_G” need not be erected so early if the measurement of the as-built information can be supported by other alternatives.

The alternative solution should provide the temporary support within the space under the steel beam. Therefore, the spatial criterion is that the physical boundary of the alternative provider should intersect with the physical space of “Scaffold_G”. Meanwhile, the temporal criterion is a time-window which should start later than the completion of the welding and inspection of the steel beam on Day 222 to ensure that the steel beam is ready for as-built information. It should also finish before the interference from “MainCon” access from Day 227 to Day 240 (See Figure 7.30).

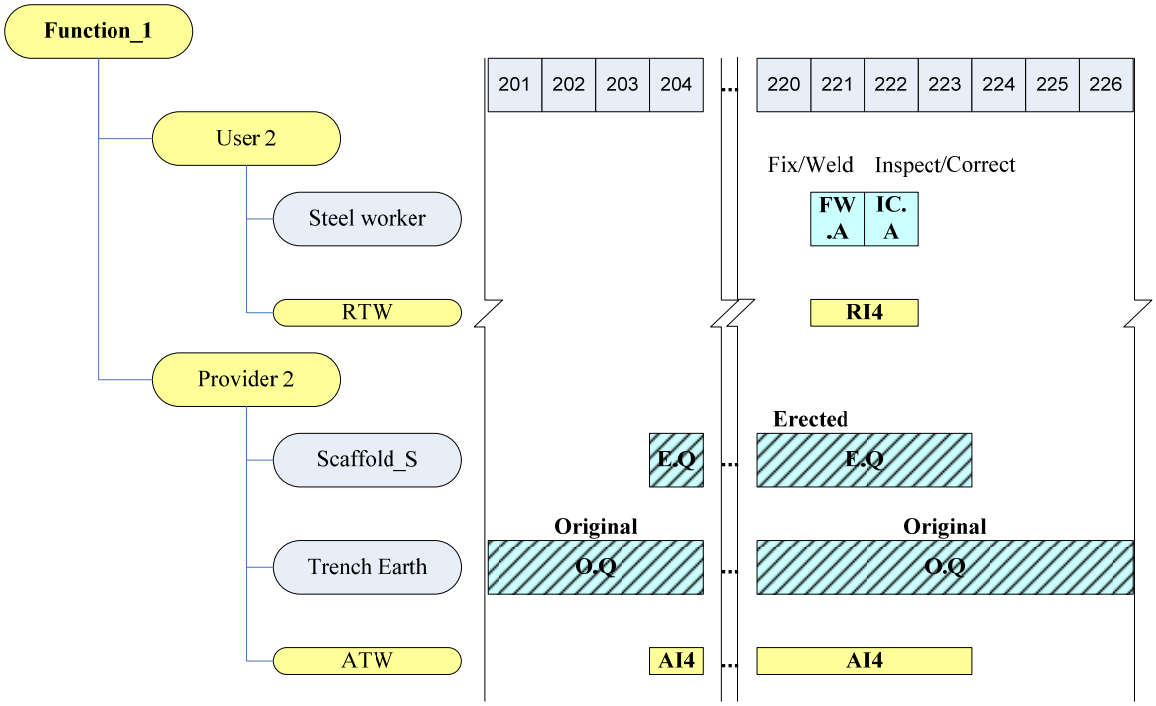



**Figure 7.32 Temporary Support Function for Steel Beam Works**

Furthermore, another scaffold “Scaffold_S” satisfying the above-mentioned spatial and temporal criteria can also be identified to provide the fourth alternative. Figure 7.32 illustrates the temporary support function “Function_2” associated with “Scaffold_S”, wherein the “Steel Workers” (function user) utilize “Scaffold_S” to install and check the steel beam. “Scaffold_S” and the underlying “Ground Earth” jointly work as the function provider.

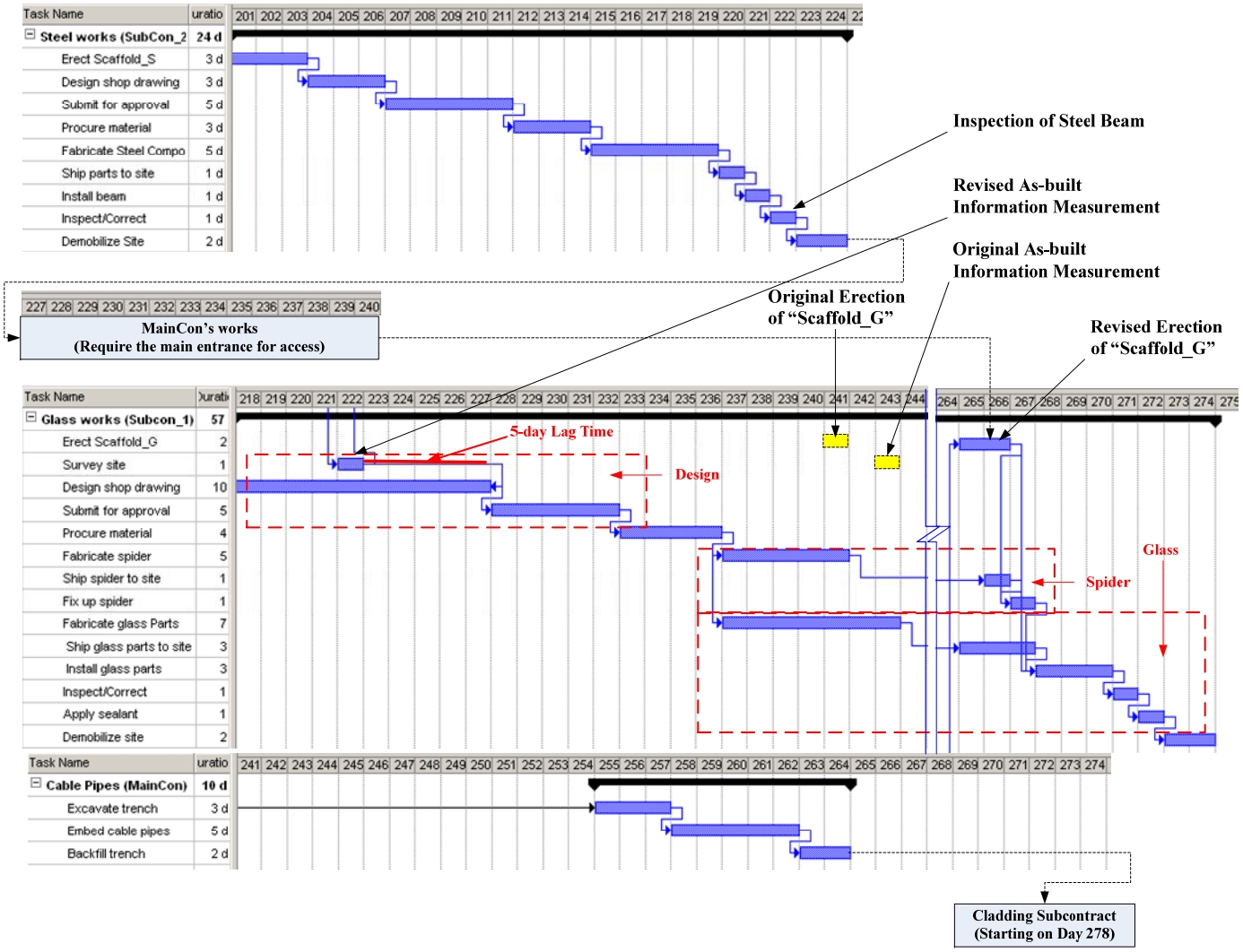



**Figure 7.33 Improved Schedule for Beam, Glass Works, and Cable Pipes**

The revised construction schedule in Figure 7.33 shows the measurement of the as-built information of the steel beam being brought forward from Day 243to Day 222 to coincide with the inspection of the steel beam. Moreover, the dismantling of "Scaffold_S" is not delayed. This also advances the shop drawing design of the glass works due to earlier acquisition of as-built information, leading to earlier fabrication of spider and glass components. Since glass fabrication is a long-lead item and the fabricated panels may need to be re-fabricated if damaged during transportation, earlier availability of the shop drawings will reduce the risk of delayed glass fabrication, leading to improved constructability of glass works.

Consequently, the erection of "Scaffold_G" that is originally planned to start on Day 241 can be postponed to Day 265 (25 days later) just after the excavated trench is backfilled so that the conflict between the erection of "Scaffold_G" and the excavation of the "Trench Earth" can be resolved. The improved schedule in Figure 7.33 shows that this alternative does not delay the installation of glass components.

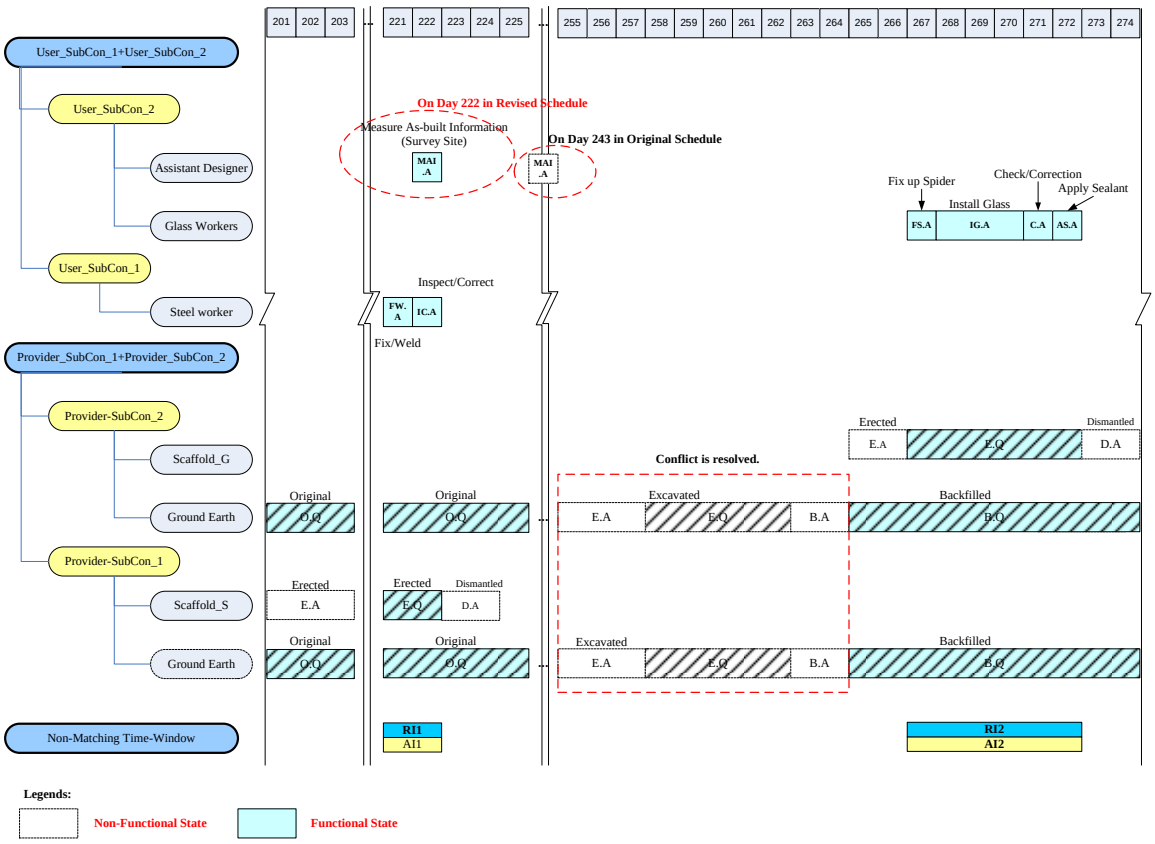



**Figure 7.34 Co-Matching Two-Users and Two-Providers for Temporary Support**

The fulfillment of the two aforementioned temporary support functionalities can be evaluated with respect to co-matching the requirement time-windows (“RI1” and “RI2”) and the availability time-windows (“AI1” and “AI2”) associated with “SubCon_1” and “SubCon_2”. Figure 7.34 shows the availability time-windows and requirement time-windows for both temporary support functions according to the revised construction schedule in Figure 7.33. Evidently, the resultant non-matching time-windows is null, indicating that the two functionalities are now available when required. Specifically, the temporary support requirement from the “Assistant Designer” (User_SubCon_2) for measuring as-built information can be fulfilled by “Scaffold_S” erected by “SubCon_2” (Provider_SubCon_1) by advancing its requirement interval from Day 243 to Day 222, as illustrated in Figure 7.34. This means that the collaboration on

resolving intermediate function requirements across trades may improve the overall executability of the construction schedules, leading to improved constructability of the project.

## 7.3 Concluding Remarks

In this chapter, the developed intermediate function analysis framework is validated against two case studies. The first case study illustrates that the schema for representing intermediate function requirement knowledge can capture the temporal logics, especially the non-precedent ones, residing in the construction methods for a complex structure. It also demonstrates that the in-progress product core model (IPPCM) together with the product oriented scheduling technique (POST) can be used for developing construction schedules from the product instead of process viewpoint. This case study also verifies that not only can the state relationships represent the construction sequences in terms of precedent relationships but they can also describe the functional interdependencies and concurrent works with respect to coincident and coupling relationships. In this way, the non-precedent relationships residing in construction schedules can be semantically represented and evaluated.

The first case study also demonstrates the value of the identification and analysis of bottleneck states. In this instance, an 8 percentage reduction of construction period was achieved. Furthermore, new state relationships inferred from existing state relationships can help planning engineers analyze the feasibility of advancing bottleneck states, leading to selection of alternative construction methods. Additionally, this case study also demonstrates the values of 4D simulation for developing and explaining construction

sequences and for facilitating intermediate function analysis as well as for preventing potential collisions.

The second case study demonstrates that the co-matching between multiple requirement time-windows (users) and multiple availability time-windows (providers) can enhance constructability with respect to saving cost on extra scaffold and not delaying the delivery of site to the downstream trade. It also illustrates the benefits from making the intermediate function requirements transparent among the trades. This case study also implies that the co-matching analysis provides a platform among the relevant trades for collaboratively evaluating their segmented schedules, leading to improved overall executability. Additionally, the second study indicates that the 4D simulation can facilitate locating alternative engineering solutions for resolving unfulfilled intermediate functions.

# CHAPTER 8 CONCLUSIONS AND RECOMMENDATIONS

## 8.1 Reviews of Intermediate Function Analysis Framework

The AEC community still lacks a modeling tool to represent and evaluate functional construction requirements. Additionally, the representation of functional construction requirements is also restricted by the previous definition of the construction state concept, which does not distinguish active phases and quiescent phases. Furthermore, the concurrent relationships between construction activities are seldom semantically represented in traditional CPM schedules, which also affect the analysis of functional construction requirements. The aforementioned inadequacy may greatly affect the constructability of facility project with respect to the executability of its construction schedule. Therefore, the present study focuses on developing an intermediate function analysis framework to improve constructability of facility projects.

Specifically, the present study proposes the intermediate function concept and the corresponding semantic model to represent a class of functional construction requirements arising from supporting the construction processes and maintaining the temporary stability of in-progress structures. Meanwhile, it also develops the component state concept in the context of in-progress product modeling. The component state concept is indispensable for representing an intermediate function, and also makes it possible to evaluate the fulfillment of intermediate function from both temporal and spatial perspectives. Moreover, the schema for representing intermediate function requirement knowledge, the information integration framework, and four intermediate function analysis methodologies are also developed for implementing intermediate function analysis. In this way, the analysis results can aid planning engineers to improve

the executability of construction schedules, leading to better constructability of facility projects.

The proposed analysis framework can be scaled to an entire AEC project. Actually, the modelling scope of a facility project can be flexibly determined by designers and constructors according to the nature of the constructability problems, while the decomposition of product components, process activities, resource items, and even intermediate functions can be granulated to different levels according to the required accuracy of the constructability analysis. No matter what granularity and scope of the analysis model are needed, the association relationships between the entities in the various project perspective models should be created and maintained in the system.

However, the product, process, and intermediate function models as well as other project perspective models are developed by different constructors or designers at different times with different granularities. In this connection, the accuracy of the analysis result may be affected by the coarseness of the models. In some situation, the lack of refined product model, 3D space model, or specification of crucial construction methods may render the analysis of some intermediate function requirements unworkable. Meanwhile, the intermediate function analysis may also be affected by the unsynchronized input of project information models developed by different trades due to the fragmented and dynamic nature of construction project management as well as the inability to foresee far in advance actual collaboration conditions. In particular, some component state information may not be available when required especially in early project phases. It also may occur that an intermediate function requirement is identified too late to be realized economically due to lack of timely project model information.

Taking the industry experts' suggestions into account, some of the difficulties arsing from undesirable granularity and availability of various project perspectives can be partially overcome by improving collaborative environment among the trades. For example, a part of the product model originally developed by "Constructor A" only to the work zone granularity may be further decomposed according to the analysis request from "Constructor B". The main contractor can play the collaborator role for coordinating the project information management with respect to controlling the model granularity and synchronizing the input of different models and integration. Additionally, a distributed project information management system that links the different participants via internet may help the intermediate function analysis, and the prevailing project modelling standard like Industry Foundation Class (IFC) may facilitate the implementation of such a system.

In this connection, this research also helps AEC practitioners understand in-depth the potential problems arsing from fragmented management of a facility project from the construction requirement viewpoint, and also provides theoretical framework to affirm the common sense that better collaboration among trades often lead to improved constructability. From the viewpoint of synchronizing the modelling information among trades, when a constructor determines the granularity and the input time of the model information, he cannot only consider his internal requirements for constructability analysis but should also take into account the analysis needs from his fellow constructors.

## 8.2 Conclusions

### 8.2.1 In-Progress Product Modeling with Component State

The In-Progress Product Core Model (IPPCM) provides a modeling methodology to represent the in-progress configuration of a facility product. This modeling method extends the traditional product decomposition hierarchy model using a component state network, which comprises a collection of component states and the interval-to-interval state relationships between the states. Additionally, the Product-Oriented Scheduling Technique (POST) provides an approach to derive the temporal attributes of a component state from the associated construction process schedule. This implies that In-Progress Product Core Model can act as a tool to plan construction program from the product-oriented viewpoint instead of process perspective.

In a component state network, the construction life cycle of product component can be depicted by its state chain, while the functional interdependencies and concurrent transitions between two in-progress product components can be described using cross-component state relationships. This means that the concurrent relationships in construction schedules can be semantically represented with respective various types of state relationships. It also implies that new state relationships can be inferred from the known state relationships, which is elaborated in the bottleneck state analysis of first case study. Moreover, compared with the previous construction state concept, the duration of component state is divided into an active phase and a quiescent phase to better describe the requirement and availability conditions of intermediate functionalities.

### 8.2.2 Semantic Model of Intermediate Function

The present research presents a semantic representation model to describe an intermediate function in five layers, namely description perspective, constituent component, component state, state attribute, and attribute relationship. This makes it possible to evaluate intermediate function requirements from both temporal and spatial perspectives. Meanwhile, by using the developed semantic model, the engineers in different domains/trades can represent and communicate their intermediate function in a consistent format. Thus, the intermediate function concept and the semantic representation model are two fundamental issues for intermediate function analysis.

### 8.2.3 Schema for Representing Intermediate Function Requirement Knowledge

The present study also provides a schema to represent intermediate function requirement knowledge from three perspectives, namely construction life cycle of single product component using *state chain type*, functional interdependencies between two in-progress product components using *state interaction type*, and availability condition of an intermediate functionality provided by a group of product components using *intermediate function availability type*. The representation schema uses two product-oriented constructs, namely component type and state type, and four categories of interval-to-interval relationships, which are precedent, coincident, coupling, and disjoint.

This representation schema enriches the terminologies to depict the temporal relationship types other than only precedent type. This knowledge representation schema may help planning engineers capture more temporal logics residing in construction methods and then incorporate these logics into intermediate function analysis. In this way,

the executability of construction schedules can be better evaluated from a broader perspective not restricted merely to traditional precedence relationships. Meanwhile, reusing these three types of intermediate function requirement knowledge can facilitate intermediate function analysis. Specifically, the construction life cycle of a product component can be initialized from the associated state chain type, while the state relationships between the state chains can be created by using the state interaction knowledge, and the functional state package of an intermediate function can be derived from the associated availability condition type.

### 8.2.4 Integration Framework for Intermediate Function Analysis

The present study develops an information integration framework for intermediate function analysis. This framework integrates five modeling perspectives necessary for intermediate function analysis. These five perspectives are product, process, resource, intermediate function, and space. In this framework, the component state network acts as the reference kernel to link product, process, intermediate function, and space models. Specifically, the process model links with the component state network through work packages and references resource model through performer packages, while the intermediate function model associates with the component state network through the requirement and functional state packages. Additionally, a 3D space model references the corresponding component state network via the state space attributes of the component states. Based on this framework, the temporal and spatial attributes can be derived, and then the analysis approaches can use these derived attribute to detect unfulfilled

intermediate function requirements and to find bottlenecks that delay the availability of the required intermediate functionalities.

### 8.2.5 Intermediate Function Analysis Methodologies

Four methodologies have been developed for analyzing intermediate function requirements from different aspects. The first methodology quantitatively evaluates the availability of the temporal interaction between the user and the corresponding provider by matching the requirement and availability time-windows, while the second analysis tool diagnoses the spatial interaction between them by evaluating the derived spatio-temporal interaction matrix against the corresponding spatio-temporal criterion matrix.

In this way, not only the unfulfilled intermediate function requirements can be detected with respect to unavailable temporal or spatial interaction, but the unavailability time-windows and the undesirable component states as well as the unsatisfactory topological relationships are also identified. This information can help planning engineers to identify the cause of the unfulfilled intermediate function requirements, and also support their decision-making on adjusting construction schedules or changing construction methods. Additionally, the spatial interaction analysis also implies that non-intersection topologies (like “meet”) can be incorporated and then evaluated for detecting unfulfilled spatial requirements in construction schedule. Many previous time-space studies focus on detecting spatio-temporal conflicts arising from only temporal collisions (temporal intersection relationships).

The third analysis methodology extends the temporal interaction analysis between a single user and a single provider to multiple users and providers for a set of

substitutable/compatible intermediate functionalities. The analysis result shows that unfulfilled requirement in a single intermediate function can be resolved by one or several alternative providers. This analysis methodology can aid designers and constructors in collaboratively resolving their intermediate function requirements, leading to saving on temporary facilities and earlier deliveries. Additionally, this analysis also reminds construction planners that some intermediate function requirements may cause extra expenditure if they are not resolved in the earlier construction stages, especially when the compatible intermediate functionalities are being provided by other trades.

The fourth and last analysis methodology is used for identifying bottleneck states. The present study reveals that a bottleneck state determines the earliest availability of the associated intermediate functionality. Sometimes, the commencement of a construction activity is constrained by the availability of an intermediate functionality, implying that if the availability of the required intermediate functionality can be advanced the construction period can be reduced. In this case, the bottleneck analysis can help planning engineers to adjust construction schedules for early delivery of project. Moreover, the analysis result can also guide them to proactively control the completion of the upstream works that may delay the downstream bottleneck states, especially when these bottleneck states constrain some critical activities. Additionally, the bottleneck state analysis also provides valuable insights for project managers to select alternative construction methods for early realization of the required intermediate functionalities.

### 8.2.6 Research Prototype

The research prototype 4D Intermediate Function Analysis Tool (4D-iFAST) is developed for implementing the integration framework and the analysis methodologies. This prototype indicates that the proposed intermediate function analysis can be vastly facilitated by using the information technologies to manage the data and to automatically detect the unfulfilled intermediate functions. Additionally, the prototype also implies that the intermediate function analysis can be facilitated by using 4D simulation by visualizing the in-progress configuration of facility product and facilitating identification of alternative engineering solutions for realizing the required intermediate functionalities. Moreover, this prototype further implies that it can be used as a collaborative center for the project participants to make transparent and communicate their intermediate function requirements, to share their intermediate function requirement knowledge, and to jointly resolve their intermediate function requirements.

### 8.2.7 Existence Vector and Boolean Operations

The existence vector concept and the Boolean operations have been developed for representing both temporal intervals and discontinuous time-windows. The present study defines three fundamental Boolean operations, namely Boolean union, intersection, and cut, on two existence vector operands, and also develops another two additional Boolean operations “multiple-union” and “multiple-intersection” on a set of existence vectors. The existence vector together with the Boolean operations makes it possible to evaluate temporal and spatial interactions without having to conduct 4D simulation. Additionally, the temporal data representation in intermediate function analysis can be simplified by

using existence vector as a representation format, and the Boolean operations on existence vectors can also speed up the intermediate function inference engine and the 4D simulation engine since the computation of Boolean values are much faster than the computation of comparing integers or floats required by the traditional interval algebra computations.

## 8.3 Limitations

In the course of the present research, some limitations have been observed and summarized as follows.

### 8.3.1 Timely Awareness on Intermediate Function Requirements

A major challenge to implement the proposed intermediate function analysis lies in that some AEC companies may lack of awareness on construction requirement management. Compared with most software developers who regard the requirements of a software project as an independent management issue, many AEC practitioners distribute their construction requirements in different management facets like site layout, construction method selection, and construction scheduling. This means that some crucial intermediate function requirements required by the downstream trades may be neglected by the upstream trades. To achieve an executable schedule, each participant should adequately and in a timely fashion make its intermediate function requirements transparent to other parties so that all participants are aware of each other's requirements. The value of the intermediate function analysis would be affected if the intermediate

function requirements cannot be represented and communicated with other trades in a timely fashion.

### 8.3.2 Limitations Pertaining to Modeling Spatial Interaction

In general, the availability of an intermediate functionality not only depends on the in-progress states of each provider components, but should also rely on the structure of an in-progress provider system. In this connection, the temporal topological relationships between the in-progress provider components should also be the necessary availability conditions for the intermediate functionality, but this modeling facet has not been studied in detail in the present study due to limitation in research time. However, the semantic model has provided a good framework to incorporate future research findings. This can be an important area of research in the future.

On the other hand, the present study focuses on analyzing the temporal meet topological relationships between the function user and the function provider in the context of analyzing temporary support. There may be other types of spatial interaction between them. Some of these “non-meet” spatial interactions may arise from providing operation space for manipulating tools and equipment, providing safe workspace for labor, and preventing damages and hazards. Accordingly, the rules for defining spatio-temporal criterion matrixes should be further studied.

Additionally, the present space model may not have incorporated all space requirements that may occur during construction. Specifically, the space requirement related to work protection and safety management should be modeled in a space system in order to enrich the spatial interaction residing in intermediate functions. For example,

the space affected welding flame and the space filled with the volatile vapor from painting should be modeled for evaluating the fulfillment of the safety requirement.

### 8.3.3 Limitations Relating to Prototype

The bottleneck state analysis and the co-matching multi-users and multi-providers analysis methodologies have been developed in the current research, but have not been programmed in the prototype mainly due to the limited time. However, the developed prototype can be extended to incorporate these analysis methodologies in the future.

Another limitation of the present prototype is that the topological relationship computation algorithm programmed in the present prototype can only produce "approximate" topological relationship, which is the topological relationship between the box boundaries of two space entities instead of the accurate topological relationship between two space entities.

Additionally, after the POST derives the temporal attributes for each component state, the interval-to-interval relationships between the states can be computed according to their temporal attributes. These computed state relationships can be compared against the state relationships derived from the state interaction knowledge. The inconsistencies indicate potential conflicts in the process schedules. This analysis approach can be further studied and put into the intermediate function analysis framework, which can enhance the prototype development in the future.

## 8.4 Recommendations for Future Works

The intermediate function analysis methodology proposed in the present research may be extended with respect to the following research issues, taking the aforementioned limitations into account.

### 8.4.1 Intermediate Function Modeling

As stated in the preceding section, one major future work is to improve the semantic representation model of the intermediate function to enhance the analysis of spatial interaction. Meanwhile, the criterion rules of defining the spatio-temporal criterion matrix should be further explored for representing the non-meet criteria.

### 8.4.2 Exploration of Feasibility to Describe State of Product Subsystem

The present research only used the component state concept to describe the changing engineering behaviors of an element product component in the lowest level of the product decomposition hierarchy. Further research may be carried out to extend the component state from describing the in-progress engineering behaviors of element components to include product subsystems in the higher decomposition levels. Accordingly, the temporal relationships between the extended states of product subsystems may also be explored.

### 8.4.3 Feasibility of Advancing Bottleneck State

The first case study in Chapter 7 implies that new state relationships can be inferred from the known state relationships according to some inference rules. For

example, Based on a “Meet” relationship and a “ContainedBy” relationship predefined in the component state network, a new “Meet/Before” relationship can be generated. In the future, more inference rules should be explored. In this way, by comparing the state relationships derived from the temporal attributes with the generated state relationships, more conflicts in construction schedules can be detected.

### 8.4.4 Automatic Resolution of Unfulfilled Requirements

One important future research task is to develop the algorithm for automatically resolving the unfulfilled intermediate function requirements. In the present research stage, it can be intuitively perceived that such an algorithm should be non-polynomial hard. The proof of the non-polynomial hard nature can be conducted mathematically in the future works. The automatic resolution of the unfulfilled requirements may be achieved using evolutionary algorithms. A likely possibility is the use of genetic algorithms, and annealing methods can also be explored.

### 8.4.5 Further Validation of the Analysis Framework against Other Types of Projects

In the future, the developed intermediate function analysis framework should be further validated against more functional construction requirements from other types of projects. Processing and off-shore projects are two possible fields that can be explored.

# REFERENCES


Allen, J. F. (1984) “Towards a general theory of action and time”, Artificial Intelligence, 23, 123-154.

Angelides D.C. (1999) “Project management and good technical and business practices”, Journal Of Management In Engineering, 15(3), 78-88.

Akinci, B., Fischer, M, and Kunz, J. (2002) “Automated generation of work spaces required by construction activities”, Journal of Construction Engineering and Management, 128(4), 306-315.

Akinci, B., Fischer, M, Levitt, R., and Kunz, J. (2002) “Representing work spaces generically in construction method models”, Journal of Construction Engineering and management, 128(4), 296-305.

Akinci, B., Fischer, M., Levitt, R., and Carlson R. (2000) “Formalization and automation of time-space conflict analysis.” CIFE Working Paper #59, Stanford University. Website: http://www.stanford.edu/group/cife/online.publication/w059.pdf., Acceessed on August 2000.

Akinci, B., Fischer, M., Levitt, R., and Carlson, R. (2002) “Formalization and automation of time-space conflict analysis”, Journal of Computing in Civil Engineering, 16(2), 124-134.

Alan, R.J., and Wilfred, M. (1971) “CI/SfB project manual : organizing building project information, incorporating the authoritative United Kingdom version of the international SfB classification system as it applies to project information”, Imprint London , Architectural Press for RIBA Services Ltd.

Allott, T. (1998) “Common arrangement of work sections for building works”. 2nd edition, Imprint: The Construction Project Information Committe, 1998.

Anderson, S.D., Fisher D.J., and Rahman, S.P. (2000) “Integrating constructability into project development: a process approach”, Journal of Construction Engineering and Management, ASCE, 126(2), 81-88.

Ballard, G. (2000). “Positive vs negative iteration in design” Proceedings of IGLC-8, Brighton, U.K.

Björk, B.C. (1992) “A unified approach for modeling construction information.” Building and Environment, 27(2), 173-194.

Björk, B.C. (1994) “RATAS project - developing an infrastructure for computer-integrated construction.” Journal of Computing in Civil Engineering., ASCE, 8(4), 401-419.

Björk, B.C. (1989) “Basic structure of a proposed building product model.” Computer-Aided Design, vol. 21, no. 2, pp. 71-78.

Björk, B.C. and Penttilä, H. (1989) “A scenario for the development and implementation of a building product model standard.” Advances in Engineering Software, 11(4), 176-187.

Björk, B.C. and Penttilä, H. (1991) “Building product modeling using relational databases, hypermedia software and CAD systems.” Microcomputers in Civil Engineering, 6(4), 267-279.

Bobrow, D.G. (1984) “Qualitative reasoning about physical systems: an introduction”, Artificial Intelligence, 24(1-3), 1-5.

Bogus, S.M., Molenaar, K.R., and Diekmann, J. E. (2005). “Concurrent engineering approach to reducing design delivery time”, Journal of Construction Engineering and Management, 131(11), 1179-1185.

Braunsperger, M. (1996) “Designing for quality - An integrated approach for simultaneous quality engineering”, Proceedings of the Institution of Mechanical Engineers Part B-Journal of Engineering Manufacture, 210(1), 1-10.

Chakrabarti, A. (1998) “Supporting two views of function in mechanical designs”, Proceedings, Workshop on Functional Modelling and Teleological Reasoning, 15th National Conference on Artificial Intelligence (AAAI-98), Madison, WI, USA; Edited by J. Sticklen.

Chakrabarti, A. and Bligh, T.P. (2001) “A scheme for functional reasoning in conceptual design”, Design Studies, 22, 493-517.

Chakrabarti, A., and Blessing, L. (1996) “Special Issue: Representing functionality in design”, Artificial Intelligence for Engineering Design, Analysis and Manufacturing, 10(5), 251–253.

Chakrabarti, A., and Bligh, T.P. (1996) “An approach to functional synthesis of solutions in mechanical conceptual design. Part III: Spatial configuration”,, Research in Engineering Design 8, 116–124.

Chandrasekaran, B., and Josephson, J.R. (2000) “Function in device representation”, Engineering with Computers, 16, 162–177.

Chang, A.S., and Ibbs, C.W. (1998) “On-Call Contracting Strategy and Management”, Journal of Management in Engineering, 14(4), 35-44.

Chau, K. W.; Anson, M.; and Zhang, J. (2004) “Four-dimensional visualization of construction scheduling and site utilization”, Journal of Construction Engineering and Management, 130(4), 598–606.

Chen, Q., Chua, D.K.H., and Song, Y.B. (2003) “Information flow integrated process modeling”, Proceedings of the International Group of Lean Construction 11th Annual Conference, Virginia Tech, Blacksburg, Virginia, USA, July 22 - 24, 2003.

Cherneff, J., Logcher, R., and Sriram, D. (1991) “Integrating CAD with Construction-Schedule Generation”, Journal of Computing in Civil Engineering, ASCE, 5 (1), 65-84.

Chevallier, N., Russell, A.D. (1998) “Automated Schedule Generation”, Canadian Journal of Civil Engineering, 25(6),1059-1077.

Chittaro, L., Guida, G., Tasso, C. and Toppano, E. (1993) “Functional and teleological knowledge in the multimodeling approach for reasoning about physical systems: a case study in diagnosis”, IEEE Transactions on Systems, Man and Cybernetics, 23(6), 1718-1751.

Chua, D.K.H., and Song, Y.B. (2003) “Component State Criteria Representation to Incorporate Construction Program Knowledge for Constructability Analysis”, Proceedings of 2003 Construction Research Congress, March. 19-21, Hawaii, 2003.

Chua, D. K. H. and Song, Y.B. (2001) “Component State Model and its Application in Constructability Analysis of Construction Schedule”, Proceedings of the 8th International 12. Conference on Civil Structural Engineering Computing, Eisenstadt, Austria 19-21, Sept. 2001.

Chua, D.K.H. and Shen, L. (2005) “Key constraints analysis with integrated production scheduler”, Journal of Construction Engineering and Management, 131(7), 753-764.

Chua, D.K.H. and Song, Y.B. (2003) “Component State Model and its Application in Constructability Analysis of Construction schedule”, Advances in Engineering Software, 34(11-12), 671-681.

Chua, D.K.H., Bok, S.H., and Song, Y. (2003) “Web-based Virtual Construction: Internet-based Collaboration in design and Construction of AEC Projects”, Technical Report for RP.R-264-000-058-112, Department of Civil Engineering, National University of Singapore, March 2003.

Chua, D.K.H., Shen, L., and Bok, S.H. (2003) “Constraint-based planning with integrated production scheduler over internet”, Journal of Construction Engineering and Management, 129(3), 293-301.

Construction Industry Institute (1986a) “Constructability: A primer” Publication of 3-1, The University of Texas at Austin.

Construction Industry Institute (1986b) “Constructability improvement during conceptual planning” SD-4, Austin, Texas.

Construction Industry Institute (1986c) “Constructability improvement during engineering and procurement” SD-5, Austin, Texas.

Construction Industry Institute (1987a) “Constructability concepts file” Publication of 3-3, Bureau of Engineering and The University of Texas at Austin.

Construction Industry Institute (1987b) “Guidelines for Implementing A Construcability Program” Publication of 3-2, The Construction Industry Institute and The University of Texas at Austin.

Construction Industry Institute (1988). “Constructability improvement during field operations” SD-34, Austin, Texas.

Construction Industry Institute (1992) “Project-Level Model and Approaches to Implement Constructability” SD-82, Austin, Texas.

Construction Industry Institute (1993) “Constructability: Program Assessment and Barriers to Implementation” SD-85, Austin, Texas.

Construction Management Committee of the ASCE Construction Division (1991), “Constructability and constructability programs: White paper”, Journal of Construction Engineering and Management, ASCE, 117(1), 67-89.

Crawford, M., Cann, J., and O'Leary, R. (1997) “Uniclass: unified classification for the construction industry”. Publication and Imprint: Royal Institute of British Architects, London.

Cysneiros L.M., and Yu, E. (2004) “Non-Functional Requirements Elicitation”, edited by Sampaio do Prado Leite J. C. and Doorn, J. H., Kluwer Academic Publishers, Boston, 115-138.

Dawood, N and Marasini, R. (2000) “Optimisation of stockyard layout for precast building products industry”, Editors Li, H., Shen,Q., Scott D., and Love, P.E.D., Implementing IT to obtain a competitive advantage in the 21st century, INCITE’2000 conference, Hongkong, 626-628.

de la Garza, J.M., Alcantara, P., Jr. and Kapoor, M., and Ramesh, P.S. (1994). "Value of concurrent engineering for A/E/C industry", Journal of Management in Engineering, 10(3), 46-55.

Deng, Y.M. (2000) "Functional design of mechanical products: Design model and modeling framework", PhD Thesis, Nanyang Technological University, Singapore.

Deng, Y.M. (2002). "Function and representation in conceptual design", Artificial intelligence for Engineering Design, Analysis, and Manufacturing, 16, 343-362.

Deng, Y.-M., Britton, G.A., & Tor, S.B. (2000c) "Constraint-based functional design verification for conceptual design", Computer Aided Design, 32(14), 889–899.

Deng, Y.M., Britton, G.A., and Tor, S.B. (1998) "A design perspective of mechanical function and its object-oriented representation scheme", Engineering with Computers, 14(4), 309–320.

Deng, Y.-M., Tor, S.B., & Britton, G.A. (2000b) "A dual-stage functional modeling framework with multi-level design knowledge for conceptual mechanical design", Journal of Engineering Design, 11(4), 347–375.

Deng, Y.-M., Tor, S.B., and Britton, G.A. (2000a) "Abstracting and exploring functional design information for conceptual mechanical product design", Engineering with Computers, 16(1), 36–52.

Dhillon, B.S. (2002) "Engineering and technology management tools and applications", Imprint London, Artech House, 193-210.

Dzeng, R.J. and Tommelein, I. D. (1995) "Case-Based Scheduling Using Product Models" Proceedings of the Second Congress held in conjunction with A/E/C Systems '95 held in Atlanta, Georgia, J. P. Mohsen, (editor), June 5-8, 1995, 163-170.

Dzeng, R.J. and Tommelein, I. D. (1997) "Boiler Erection Scheduling Using Product Models and Case-Based Reasoning" Journal of Construction Engineering and Management, Vol. 123(3), 338-347.

Egenhofer, M., and Franzosa, R. (1990). "A Mathematical Framework for the Definition of Topological Relationships" The 4th International Symposium on Spatial Data handling, Zurich, Switzerland, 803-813.

Eldin, N., and Hikle, V. (2003) "Pilot study of quality function deployment in construction projects", Journal of Construction Engineering and Management, 129(3), 314-329.

Eldin, N.N. (1997). “Concurrent engineering: A schedule reduction tool”, Journal of Construction Engineering and Management, 123(3), 354-362.

Fischer, M., and Aalami, F. (1996) “Scheduling with computer-interpretable construction method models”, Journal of Construction Engineering and Management, ASCE, 122(4), 337-347.

Fischer, M., and Froese, T. (1996) “Examples And Characteristics Of Shared Project Models.”, Journal of Computing in Civil Engineering , 10(3),174-182.

Fisher, D.J., O’Connor, J.T. (1991) “Constructability for piping automation - Field operations”, Journal of Construction Engineering and Management-ASCE 117(3), 468-485.

Fisk, E.R. (2003) “Construction project administration”, Imprint Upper Saddle River, N.J. : Prentice Hall, 7th Edition, 406-422.

Froese, T. (1996) “Models of Construction Process Information”, Journal of Computing in Civil Engineering, 10(3), 183–193.

Guo, S.Y. (2002) “Identification and resolution of work space conflicts in building construction”, Journal of Construction Engineering and Management, 128(4), 287-295.

Hawkins, P.G. and Woollons, D.J. (1998) “Failure modes and effects analysis of complex engineering systems using functional models”, Artificial Intelligence in Engineering, 12(4), 375-397.

Hawkins, R., Sticklen, J., McDowell, J.K., Hill, T. and Boyer, R. (1994) “Function-based modelling and troubleshooting”, Applied Artificial Intelligence, 8, 285-302.

Hayes, C.C (1995) “Use of function information to create redesign suggestions”, Proceedings of the 8th Florida AI Research Symposium (FLAIRS-95), Special Track on Reasoning about Function, Melbourne, FL, 309-313.

Heesom, D., Mahdjoubi, L., and Proverbs, D. (2003) “A dynamic VR system for visualizing construction space usage”, Construction Research Congress In Construction - Wind of Change: Integration and Innovation, Proceedings of Construction Research Congress, March 19-21, 2003, Honolulu, Hawaii.

Hegazy T., and Elbeltagi E. (1999) “EvoSite: evolution-based model for site layout planning”, Journal of Computing in Civil Engineering, 13(3), 198-206.

Hendrickson, C., Au, T. (1989) “Project Management for Construction: Fundamental Concepts for Owners, Engineers, Architects and Builders”, Prentice-Hall, Englewood Cliffs, NJ.

International Alliance for Interoperability (2004) “Industry Foundation Classes IFC2x”, 2nd Edition, Website: http://www.iai-international.org/iai_international/Technical_Documents/R2x2_final/index.html, last accessed, April 19, 2004.

ISO TC184/SC4/WG4 (1992) “ISO 10303 Industrial automation systems and integration -- Product data representation and exchange Guidelines for the Development and Approval of STEP Application Protocols, Version 1.0”, Palmer, M. (editor), ISO TC184/SC4/WG4, Committee Draft N34, February 20, 1992.

ISO/DIS 10303-1 (1993) “Industrial automation systems and integration --Product data representation and exchange -- Part 1: Overview and fundamental principles, International Organization for Standardization”, March 26, 1993.

ISO/DIS 10303-11 (1992) “Industrial automation systems and integration -- Product data representation and exchange -- Part 11: Description methods: The EXPRESS language reference manual, International Organization for Standardization, August 31, 1992.

Jägbeck, A. (1994) “MDA Planner: Interactive planning tool using product models and construction methods”, Journal of Computing in Civil Engineering, 537-554.

Kitamura Y., and Mizoguchi R. (2003) “Ontology-based description of functional design knowledge and its use in a functional way server”, Expert Systems with Applications, 24(2), 153-166.

Kitamura, Y. and Mizoguchi R. (1999) “Meta-functions of artifacts”, 13th International Workshop on Qualitative Reasoning (QR-99), 136-145.

Koo, B. and Fischer, M. (2000) “Feasibility study of 4D CAD in commercial construction”, Journal of Construction Engineering and Management, 126(4), 251-260.

Krishnan, V. (1996). “Managing the simultaneous execution of coupled phases in concurrent product development” IEEE Transactions on Engineering Management, 43(2), 210–217.

Krishnan, V., Eppinger, S.D., and Whitney, D.E. (1995) “Accelerating product development by the exchange of preliminary product design information” ASME Journal of Mechanical Design, 117, 491–498.

Krishnan, V., Eppinger, S.D., and Whitney, D.E. (1997) “A model-based framework to overlap product development activities” Management Science, 43(4), 437–451.

Kumar, A. N. and Upadhyaya, S. J. (1995) “Function based discrimination during model-based diagnosis”, Applied Artificial Intelligence, 9(1), 65-80.

Kumar, A.N. and Upadhyaya, S.J. (1995) “Function based discrimination during model-based diagnosis”, Applied Artificial Intelligence, 9(1), 65-80.

Larsson, J.E. (1996) “Diagnosis based on explicit means-end models”. Artificial Intelligence, 80(1), 29-93.

Lee J. (1997) “Design rationale system: understanding the issues”, Expert, IEEE, 12(3), 78-85.

Lee, D., and Arditi, D. (2006) “Total quality performance of design/build firms using quality function deployment”, Journal of Construction Engineering and Management, 132(1), 49-57.

Lee, S. H., Peña-Mora, F., and Park, M. (2005) “Quality and change management model for large scale concurrent design and construction projects”, Journal of Construction Engineering and Management, 131(8), 890-902.

Li H., and Love, PED (1998) “Site-Level Facilities Layout Using Genetic Algorithms”, Journal of Computing in Civil Engineering, 12(4), 227-231.

Loch, C., and Terwiesch, C. (1998). “Communication and uncertainty in concurrent engineering” Management Science, 44(8), 1032-1045.

Low, S.P., and Larry, Y. (2001) “Quality function deployment in design/build projects”, Journal of Architectural Engineering, 7(2), 30-39.

Luiten, G.T., Tolman, F.P. (1997) “Automating Communication In Civil Engineering”, Journal of Construction Engineering and Management, 123(2), 113-120.

Ma, Z., Shen Q., and Zhang, J. (2005) “Application of 4D for dynamic site layout and management of construction projects”, Automation in Construction, 14(4), 512-524.

Mallasi, Z. (2005) “Dynamic Quantification and analysis of the construction workspace congestion utilizing 4D visualization”, Automation in Construction (2005), Article In Press, Accepted 30 August 2005, Available online 13 December 2005.

Mallasi, Z. and Dawood. N. (2001) “Assessing space criticality in sequencing and identifying execution patterns for construction activities using VR visualizations” ARCOM Doctoral research workshop: Simulation and modeling in construction, October 2001, Edinburgh University, UK, 22-27.

Mallasi, Z. and Dawood. N. (2002). “Registering Space Requirements of Construction Operations Using Site-PECASO Model.” Proceedings of CIB w78 2002 Conference on Distributing Knowledge in Buildings, 2002, Aarhus School of Architecture, Denmark, Vol. 2, 83-90.

Mallon, J.C. and Mulligan, D.E. (1993) “Quality function deployment - System for meeting customers’ needs”, Journal of Construction Engineering and Management, 119(3), 516-531.

McKinney K., and Fischer, M. (1997) “4D analysis of temporary support”, Computing in Civil Engineering, Proceedings of the Fourth Congress held in conjuction with A/E/C Systems ’97 in Philadelphia, PA, June 16-18, 1997, 470-476.

Messner, J.I., Yerrapathruni, S.C. Baratta,A.J. and Riley, D.R. (2002) “Cost and schedule reduction of nuclear power plant construction using 4D CAD and immersive display technologies”, Proceedings of the International Workshop on Information Technology in Civil Engineering, November 2-3, 2002, Washington, D.C. 136-144.

Michael, W.R., and Jeffrey, S.R. (1993) “Project-Level Model Process for Implementing Constructability”, Journal of Construction Engineering and Management, 119(4), 813-831.

Morad, A.A. and Beliveau, Y.J. (1994) “Geometric-Based Reasoning System for Project Planning”, Journal of Computing in Civil Engineering, ASCE, 8 (1), 52-71.

Mukhopadhyaya, A.K. (2003) “Value engineering: concepts, techniques and applications”, Imprint New Delhi, Sage.

Navinchandra D, Sriram D, Logcher RD (1988) “GHOST: Project Network Generator”, Journal of Computing in Civil Engineering, ASCE, 2(3), 239-254.

Nielsen, N.P.H., and Holmstrom, J. (1995) “Design for speed - a supply chain perspective on design for manufacturability”, Computer Integrated Manufacturing Systems, 8(3), 223-228.

O’Connor, J.T., and Davis, V.S. (1988) ‘‘Constructability improvement during field operations’’ J. Constr. Eng. Manage., 114(4), 548–564.

O’Connor, J.T., Rusch, S.E., and Schulz M.J. (1987) “Constructability concepts for engineering and procurement.”, Journal of Construction Engineering and Management, ASCE, 113(2), 235-247.

Open CASCADE S.A.S. (2006) “Industrial Open Source alternative to proprietary 3D modeling kernels”, Website: http://www.opencascade.org/occ/, accessed on June, 2006.

Peña-Mora, F., and Li, M. (2001) “Dynamic planning and control methodology for design/build fast-track construction projects”, Journal of Construction Engineering and Management, 127(1), 1-17.

Pham, D.T., and Dimov, S.S. (1998) “An approach to concurrent engineering”, Proceedings of the Institution of Mechanical Engineers Part B - Journal of Engineering Manufacture, 212 (1), 13-27.

Prasad, B. (1996). Concurrent engineering fundamentals: Integrated product and process organization, Prentice Hall, Upper Saddle River, N.J.

Qian, L., and Gero, J.S. (1996) “Function-Behavior-Structure paths and their roles in analogy-based design”, Artificial Intelligence for Engineering Design, Analysis, and Manufacturing, 10(5), 289-312.

Ranky, P.G. (1994) “Concurrent engineering and enterprise modeling”, Assembly Automation, 14(3), 14-21.

RIBA Enterprises, “NBS Building features”, website: http://www.thenbs.com/products/nbsBuilding/buildingFeatures.asp, accessed on Feb, 2006.

Riley, D.R., and Sanvido, V.E. (1995). “Patterns of construction-space use in multistory buildings” Journal of Construction Engineering and Management, ASCE, 121(4), 464-473.

Riley, D.R., and Sanvido, V.E. (1997). “Space planning method for multistory building construction” Journal of Construction Engineering and Management, ASCE, 123(2), 171-180.

Russell, J.S., and Gugel, J.G. (1993) "Comparison of 2 Corporate Constructability Programs." Journal Of Construction Engineering And Management-Asce, 119(4), 769-784.

Russell, J.S., Gugel, J.G., and Radtke, M.W. (1994) "Comparative-Analysis Of 3 Constructability Approaches." Journal Of Construction Engineering And Management-Asce, 120(1), 180-195.

Sapuan, S.M., Osman, M.R., and Nukman, Y. (2006) “State of the art of the concurrent engineering technique in the automotive industry”, Journal of Engineering Design, 17 (2), 143-157.

Sasajima, M., Y. Kitamura, M. Ikeda, and R. Mizoguchi (1995) “FBRL: A function and behavior representation language”, in International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, 1830-1836.

Sato, Y., and Kaufman, J.J. (2005) “Value analysis tear-down : a new process for product development and innovation”, Industrial Press : Society of Manufacturing Engineers, Imprint New York, 21-42.

Shen, L., and Chua, D.K.H. (2005) “Key constraint analysis: achieve lean processes with the application of TOC”, Proceedings of the Congress, edited by Iris D. Tommelein, April 5.7, 2005, San Diego, California.

Sheppard, L. M. (2004) "Virtual Building for Construction Projects," IEEE Computer Graphics and Applications, 24(1), 6-12.

Soetanto, R., Glass, J., Dainty, A.R.J., Price, A.D.F., and Thorpe, A. (2005) “Improving the utility and value of CAD software for decision — making and design of structural frames”, Computing in Civil Engineering (2005), Proceedings of the 2005 International Conference, July 12-15, 2005, Cancun, Mexico.

Soltani, A. R., Tawfik H., and Fernando T. (2002) “A multi-criteria based path finding application for construction site layouts” Proceedings of Sixth International Conference on Information Visualization (IV'02), 779-784.

Song, Y.B., and Chua, D.K.H. (2002) “POST: Product Oriented Scheduling Technique for Constructability Analysis”, Proceedings of the International Workshop on Information Technology in Civil Engineering, Nov. 2-3, 2002, Washington, D.C., ASCE, 124-135.

Song, Y.B., and Chua, D.K.H. (2003) “COSEE: Component State Network Centric Model for Verifying Temporal and Spatial Consistence in Project Schedules”, Proceedings of the CIS W78’s 20th International Conference on Information Technology for Construction, Waiheke Island, new Zealand, 23-25, April, 2003, 325-332.

Song, Y.B, and Chua, D.K.H. (2006) “Temporal logic representation schema for intermediate function”, submitted to the Journal of Construction Engineering and Management, ASCE, in review.

Song, Y.B., and Chua, D.K.H. (2005). “Detection of Spatio-Temporal Conflicts on a Temporal 3D Space System”, Advances in Engineering Software, 36(11-12), 814-826.

Song, Y.B. and Chua, D.K.H. (2004). “Application of Industry Foundation Classes for Intermediate Function Analysis”, Proceedings of the fourth international conference on engineering computational technology, Lisbon, Portugal, 7-9 September 2004.

Song, Y.B., and Chua, D.K.H. Chang, Q.L, Bok, S.H. (2003). "Spatio-temporal consistency evaluation on dynamic 3d space system model", Proceedings of the ninth international conference on civil and structural engineering computing, 2-4 September, 2003, Egmond-Aan-Zee, The Netherlands.

Song, Y.B, and Chua, D.K.H. (2004) "An integrated project model for detecting schedule conflicts in merged schedules", White paper submitted to Infocomm Development Authority (IDA) of Singapore for the Collaborative Engineering Program, Singapore, 2004.

Song, Y.B, and Chua, D.K.H. (2005) "Detection of spatio-temporal conflicts on a temporal 3d space system", White paper submitted to Infocomm Development Authority (IDA) of Singapore for the Collaborative Engineering Program, Singapore, 2005.

Songer, A.D., Dieckmann, J., Hendrickson, W., and Flusing, D. (2000) "Situational reengineering: case study analysis", Journal of Construction Engineering and Management, 126(3), 185-190.

Stumpf, A.L., Ganeshan, R., Chin, S., and Liu, L.Y. (1996) "Object-Oriented Model for Integrating Construction Product and Process Information", Journal of Computing in Civil Engineering, 10(3), 204-212.

Syed, M.A., Li, P.S., and Zeljko, M.T. (2003) "Use of quality function deployment in civil engineering capital project planning", Journal of Construction Engineering and Management, 129(4), 358-368.

Tatum, C.B. (1989). "Improving constructability during conceptual planning", Journal of Construction Engineering and Management, ASCE, 113(2), 191-207.

Tawfik, H., and Fernando, T. (2001) "A Simulation Environment for Construction Site Planning" Proceedings of Fifth International Conference on Information Visualization (IV'01), 0199-0204.

Terwiesch, C., Loch, C. H., and De Meyer, A. (2002) "Exchanging preliminary information in concurrent engineering: Alternative coordination strategies." Organization Science, 13(4), 402–419.

Thabet, W.Y., and Beliveau, Y. J. (1994a) "Modeling work space to schedule repetitive floors in multistory buildings" Journal of Construction Engineering and Management, ASCE,120(1), 96-116.

Thabet, W.Y., and Beliveau, Y.J. (1994b) "HVLS: Horizontal and Vertical Logic Scheduler for Multistory Projects" Journal of Construction Engineering and Management, ASCE, 120(4), 875-892.

Tommelein, I. D., Levitt, R. E., and Hayesroth, B. (1992a) “Site-layout modeling - How can artificial-intelligence help” Journal of Construction Engineering and Management, ASCE, 118(3), 594-611.

Tommelein, I. D., Levitt, R. E., and Hayesroth, B. (1992b) “SIGHTPLAN model for site layout” Journal of Construction Engineering and Management, ASCE, 118(4), 749-766.

Tommelein, I. D.., and Zouein, P. P (1993) “Interactive dynamic layout planning”, Journal of Construction Engineering and Management, ASCE, 119(2), 266-287.

Ullman D.G. (1993) “The evolution of function and behavior during mechanical design”, Design Theory and Methodology, 53, 91-103.

Umeda, Y, Ishii, M., Yoshikawa, M., Shimomura, Y., and Tomiyama, T. (1996) “Supporting conceptual design based on the function-behavior-state modeler” Artificial Intelligence for Engineering Design, Analysis, and Manufacturing, 10(5), 275-288.

Umeda, Y., Tomiyama, T. and Yoshikawa, H. (1992) “A design methodology for a self-maintenance machine based on functional redundancy”, In Design Theory and Methodology-DTM'92, ASME, 317-324.

Vilain, M., Kautz, H., and van Beek, P. “Constraint propagation algorithms for temporal reasoning: a revised report”, In Readings in Qualitative Reasoning About Physzcal Systems, D. S. Weld and J. de Kleer, Eds. Morgan Kaufmann, San Mateo, California, 1990, pp. 373-381.

Vaugn, F. (1996) “3D & 4D CAD modeling on commercial design-build projects”, Computing in Civil Engineering (1996), Proceedings of the Third Congress held in conjunction with A/E/C Systems ’96 held in Anaheim, California, June 17-19, 1996, 390-396.

Waugh, L.M. (1990) “A construction planner”, Technical Report, 32, Centre for Integrated Facility Engineering, Stanford University, Stanford, CA.

Winsor, J., and MacCallum, K. (1994) “A Review of functionality modeling in design”, The Knowledge Engineering Review, 9(2), 163–199.

Yeh, I. (1995) “Construction-site layout using annealed neural network”, Journal of Computing in Civil Engineering, 9(3), 201-208.

Yeh, R.T. (1992) “Notes on concurrent engineering”, IEEE Transactions on Knowledge and Data Engineering, 4(5), 407-414.

Younker, D.L. (2003) “Value engineering: analysis and methodology”, Imprint New York, Marcel Dekker.

Youssef, M.A. (1994) “Design for manufacturability and time-to-market, Part 1: Theoretical foundations”, International Journal of Operations & Production Management, 14(12), 6- 21.

Zouein, P. P., and Tommelein, I. D. (1999) “Dynamic layout planning using a hybrid incremental solution method”, Journal of Construction Engineering and Management, ASCE, 125(6), 400-408.

Zouein, P. P., and Tommelein, I. D. (2001) “Improvement algorithm for limited space scheduling.” Journal of Construction Engineering and Management, ASCE, 127(2), 116-124.

Zozaya-Gorostiza, C., Hendrickson, C., and Rehak, D. R. (1989) “Knowledge-based process planning for construction manufacturing”, Academic Press Incorporation, San Diego, California.

# APPENDIX: PUBLICATION LIST

**Journal Papers:**

Song, Y.B. and Chua, D.K.H., "Temporal Logic Representation Schema for Intermediate Function", Journal of Construction Engineering and Management, ASCE. 133(4), 277-286.

Song, Y.B. and Chua, D.K.H. (2006). "Modeling of Functional Construction Requirements for Constructability Analysis", Journal of Construction Engineering and Management, ASCE. 132(12), 1314-1326.

Song, Y.B. and Chua, D.K.H. (2005) "Detection of Spatio-Temporal Conflicts on a Temporal 3D Space System", Advances in Engineering Software, 36(11-12), 814-826.

Chua, D.K.H. and Song, Y.B. (2003). "Application of Component State Model for Identifying Constructability Conflicts in a Merged Construction Schedule", Advances in Engineering Software, 34(11-12), 671-681.

**Conference Papers:**

Song, Y.B. and Chua, D.K.H. (2004). "Application of Industry Foundation Classes for Intermediate Function Analysis", Proceedings of the fourth international conference on engineering computational technology, Lisbon, Portugal, 7-9 September 2004.

Song, Y.B., Chua D.K.H. Chang, Q.L, Bok, S.H. (2003). "Spatio-Temporal Consistency Evaluation on Dynamic 3D Space System Model", Proceedings of the ninth international conference on civil and structural engineering computing, 2-4 September, 2003, Egmond-Aan-Zee, The Netherlands.

Song, Y.B. and Chua, D.K.H. (2003). "COSEE: Component State Network Centric Model for Verifying Temporal and Spatial Consistence in Project Schedules", Proceedings of the CIB W78's 20th International Conference on Information technology for Construction, Wainheke Island, New Zealand, 23-25, April, 2003, 32-332.

Chen, Q., Chua, D.K.H., and Song, Y.B. (2003), "Information Flow Integrated Process Modeling", Proceedings of the International Group of Lean Construction 11th Annual Conference, Virginia Tech, Blacksburg, Virginia, USA, July 22 - 24, 2003.

Song, Y.B. and Chua, D.K.H. (2002). "POST: Product Oriented Scheduling Technique for Constructability Analysis", Proceedings of the International Workshop on

Information Technology in Civil Engineering, Nov. 2-3, 2002, Washington, D.C., ASCE, 124-135.

Chua, D. K. H. and Song, Y.B. (2002). "Component State Criteria Representation to Incorporate Construction Program Knowledge for Constructability Analysis", accepted by 2003 Construction Research Congress, March. 19-21, 2003, Hawaii.

Chua, D.K.H. and Song, Y.B. (2001). "Component State Model and its Application in Constructability Analysis of Construction Schedule", Proceedings of the 8th International 12. Conference on Civil Structural Engineering Computing, Eisenstadt, Austria 19-21, Sept. 2001.

**Technical Report:**

Chua, D.K.H., Bok, S.H., Song, Y.B., "Web-based Virtual Construction: Internet-based Collaboration in design and Construction of AEC Projects", Technical Report for RP.R-264-000-058-112, Department of Civil Engineering, National University of Singapore, March 2003.