Identifying a Design Management Package to Support Concurrent Design in Building Wafer Fabrication Facilities

Ren-Jye Dzeng[1]

Abstract: Concurrent design is commonly used in building a semiconductor wafer fabrication facilities to shorten projects. Current practice in managing a design schedule involves preset milestones that represent percentages of completion. Such a simple control scheme does not provide sufficient information to support concurrent design. This study presents an analytical model that applies a cluster identification algorithm to separate the work of designing a multisystem project into management packages that support concurrent design. Tasks within a package have strong informational dependency relationships on each other, and are not suited for concurrent design. Tasks of different packages have weak dependency relationships on each other, and are suited for concurrent design. Tendering design work based on these packages may reduce the number of design interfaces between participating design firms. Possible application of the model includes the management of design schedule, design contract tendering, and design information flow.

DOI: 10.1061/(ASCE)0733-9364(2006)132:6(606)

CE Database subject headings: Construction management; Interfaces; Algorithms; Design.

Introduction

Accelerating a product's time to market offers several advantages to manufacturers in the highly competitive semiconductor wafer industry, including reduced time to recover the investment, and increased profit. According to the SIA technology roadmap (SIA 2003), for high-demand products, volume production typically continues to ramp to semiconductor wafer fabrication facilities (FABs) design capacity within 12 months. Minimizing a facility's ramp-up time requires accelerating its design, construction, and startup. When a new wafer FAB is required, typically driven by capacity need or technological innovation to reduce production costs, such as during the inevitable transition to 300 mm wafers using 130 nm processing technology at the beginning of the new millennium (Jansen 2000), delivering the FAB according to an aggressive schedule becomes critical to the success of the investment. Chasey and Merchant (2000) also identified project delivery methods associated with a compressed schedule as a key issue of research for the construction of a 300 mm FAB.

Concurrent engineering has been adopted widely to develop a new product by integrating activities and data of multiple functional departments; it has proved to be effective in reducing product development time (Turtle 1994). Several methods of concurrent design are available for developing new products. Quality Function Deployment (QFD) (Akao 1990) identifies a product's customers, and quantitatively measures their needs and weightings of these needs before setting the product design parameters accordingly. QFD can reduce the number of required engineering changes and design uncertainty, and ensure that the product meets the customers' needs.

Design for assembly (DFA) (Boothroyd and Dewhurst 1991) is a structured analysis technique that gives design teams the information they need to reduce product costs by reducing the number of parts, simplifying parts handling, and improving product assembly by carrying out analyses on a graphical chart. Following DFA analysis, manufacturing analysis (MA) (Baudin 1990) may be applied to optimize the manufacturing process of each individual component by selecting the most cost-effective process and material at an early design stage. Such cost comparison during the concept design stage promotes cost-effective component design. Failure mode and effect analysis (FMEA) (Gordon and Isenhour 1990) is a bottom-up process for analyzing potential reliability problems early in the development cycle when such issues can be relatively easily overcome, thereby enhancing the reliability of the design.

QFD, DFA, MA, and FMEA are all concerned with improving the integration and communication of information between the stages of product development. None of these methods provides a systematic approach to breaking down a product into systems. The lack of such an approach may not represent a problem in the development of a product in which the interfaces among components can be clearly and quantitatively specified. However, it is a problem for designing and constructing a wafer FAB.

A wafer FAB is a complex fabrication plant that consists of various systems, including the architecture, the structure, the mechanics, and the clean room. These systems are designed not only by multiple functional departments within a single engineering firm, but also, very often, by multiple engineering firms. Some systems (e.g., the structure) may be delivered by the design-bid-build approach, and others (e.g., the clean room) may be delivered by the design-build approach. Properly breaking down the FAB into systems and identifying the interfaces between these systems helps the owner to manage and tender the design work by reducing the number of interorganizational interfaces.

---

[1] Professor, Dept. of Civil Engineering, National Chiao-Tung Univ., 1001 Ta-Hsieu Rd., Hsinchu, Taiwan 30050, Republic of China. E-mail: rjdzeng@mail.nctu.edul.tw

Note. Discussion open until November 1, 2006. Separate discussions must be submitted for individual papers. To extend the closing date by one month, a written request must be filed with the ASCE Managing Editor. The manuscript for this paper was submitted for review and possible publication on June 3, 2003; approved on June 22, 2005. This paper is part of the ***Journal of Construction Engineering and Management***, Vol. 132, No. 6, June 1, 2006. ©ASCE, ISSN 0733-9364/2006/6-606–614/$25.00.

Steward (1981) used a matrix to represent dependency relationships or interfaces between design activities, and developed design structure matrix analysis, which also was referred to as dependency structure matrix (DSM). DSM subsequently developed broader applications in other areas such as organizational structuring based on team interfaces (Browning 1998) and sequencing the determination of design parameters with minimized iteration (Black et al. 1990; Rask and Sunnersjö 1998).

Peña-Mora and Li (2001) applied DSM in their proposed planning methodology for a design/build fast-track construction project, an integration method combining the concepts of axiomatic design, concurrent engineering, graphical evaluation and review technique (GERT), and system dynamics. Construction alternatives were evaluated by identifying coupled dependency among activities using DSM, and were selected according to the axiomatic design concept (Suh 1990, 1995). The schedule of the chosen alternative was compressed to achieve the concurrent engineering goal by overlapping certain activities based on their product rates, production reliability of upstream activity, and sensitivity to error of downstream activity.

Browning (2001) also reviewed four types of DSMs being applied in three types of systems, including designing product architecture, process structure, and organization structure. Component-based DSM models system architecture according to components and their interrelationships. Team-based DSM models organization structures based on human interactions. Activity-based DSM models processes and activity networks based on activities and their information flow and other dependencies. Parameter-based DSM models low-level relationships between design decisions and parameters.

Unlike most building projects where the architect coordinates specialty engineering/design firms, many FAB owners coordinate specialty design firms themselves because they, as manufacturers, have better knowledge about facility systems. While activity-based DSM improves design efficiency by minimizing design process iterations, a FAB owner may still encounter the challenge of how to tender design work so that more design contracts can proceed concurrently with minimal increase in management load. While the activity-based DSM minimizes the iteration of design activities, this work concentrates on the predesign phase with a higher-level view from the perspective of project owner.

Model for Concurrent Design Planning

The proposed analytical model consists of a four-step process for preplanning concurrent design, including system breakdown, system interface identification, management package identification, and concurrent design planning. The first step is to break down the design project into systems, based on a document review or an expert interview. The document review approach is suitable for designing a familiar facility for which a previous breakdown of the structure is available. For example, the civil/structure/architecture (CSA) design may include plans, elevations, landscape, interior, and the specifications of other subsystems (e.g., of a clean room of class 100). When a designer is uncertain about the breakdown, interviews with experts, such as experienced project managers or system planners, may be necessary.

The second step is to identify the design interfaces among the systems. These design interfaces determine the sequential dependency among design tasks. Experts, such as experienced project managers or system planners, are helpful in this step. The dependencies among design tasks may be represented by a matrix whose columns represent predecessors and whose rows represent successors. The matrix clearly identifies sequential relationships among the design tasks.

The third step is to separate the systems into independent groups. Fuzzy clustering analysis (FCA) (Hoppner 1999) and the cluster identification algorithm (CIA) (Kusiak and Chow 1987) constitute two methods for completing this task. Cluster analysis involves grouping objects into homogenous groups, based on some of their features. It has been applied in a wide range of areas, including biology, data reorganization, medicine, pattern recognition, groupings parts of automated systems, production flow analysis, race mixture studies, task selection, control engineering, and expert systems (Kusiak and Chow 1987).

FCA is a method for partitioning a data set into clusters or classes, such that similar data are assigned to a single cluster, whereas dissimilar data are assigned to different clusters. The fuzzy approach is suitable for clusters among which no sharp boundaries exist. Fuzzy clustering uses degrees of membership between zero and one, rather than crisp assignments of data to clusters. The method involves three steps—establishing a fuzzy similarity measurement scheme and measuring the data, calculating the similarity among the data, and clustering the data based on their similarity. The dependencies among the design tasks are often ambiguous. However, FCA provides no obvious way to represent the sequential relationships among data.

The proposed model uses the CIA. The CIA, originally developed by Kusiak and Chow (1987), is based on a binary object-feature incidence matrix, whose columns represent objects and rows represent features. The algorithm finds mutually separable clusters, in each of which objects share the same features. Unlike the FCA, with a little adaptation, the CIA can be used to process the dependency relationships among data.

In the third step, the CIA helps to group dependent systems into management packages. Systems that belong to the same package cannot be designed concurrently. Conversely, systems that belong to different packages can be designed concurrently. CIA will be detailed in a later section.

The final step is to represent the management packages in a network schedule. The schedule may be represented using the precedence diagramming method or probabilistically using programming evaluation and review technique (PERT) (Ahuja et al. 1994). When simulation analysis is desired, GERT may be used to model alternative branches of activity looping in the schedule as illustrated by Peña-Mora and Li (2001) in their reservoir excavation example.

Another simulation alternative is Petri Nets, which is a formal and graphical language that is suitable for modeling systems with concurrency (Reisig 1992). Petri Nets has been under development (Esparza and Lakos 2002) since the beginning of the 1960s, when Petri first defined the language. An effort to create an international standard for high-level Petri Nets is also under way under the so-called 7.19.3 (Petri net Techniques) project, a subproject of 7.19 (Diagrams for Software Engineering), by the standards group called ISO/IEC JTC1/SC7/WG11 (ISO/IEC JTC1/SC7 2004). Several Petri-net-based software applications, such as ALPHA/Sim (ALPHATECH 1999), are available to analyze the concurrency and resolve resource conflicts among the activities in the network. This study uses Petri nets to represent the design plan because they are widely adopted in concurrent engineering and a broad selection of commercial software is available. Readers may refer to Reisig (1992) for the fundamentals of Petri nets.

The following conceptual example illustrates each step in detail.

J. Constr. Eng. Manage. 2006.132:606-614.

System Breakdown

Assume that a design project is divided into seven systems (S_1, S_2, S_3, S_4, S_5, S_6, and S_7), based on a review of past documents and interviews with experts.

Interface Identification

Assume that the interfaces of the systems are identified and represented by a binary successor-predecessor matrix, matrix (1). Columns represent successors and rows represent predecessors. A cell with a value of one specifies that the design of the corresponding successor depends on the design of the corresponding predecessor. A cell without a value indicates that the corresponding successor does not depend on the predecessor. For example, matrix (1) states that S_1 should be designed before S_2

$$A = \begin{array}{c|ccccccc|c} & S_1 & S_2 & S_3 & S_4 & S_5 & S_6 & S_7 & D_s \\ S_1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 7 \\ S_2 & & 1 & & & & & & 1 \\ S_3 & & 1 & 1 & & & & & 2 \\ S_4 & & & & 1 & & & & 1 \\ S_5 & & & & & 1 & 1 & & 2 \\ S_6 & & & & & & 1 & & 1 \\ S_7 & & & & & & & 1 & 1 \\ D_p & 1 & 3 & 2 & 2 & 2 & 3 & 2 & \end{array} \tag{1}$$

Management Package Identification

After the interfaces among the systems have been identified, the CIA must be used to divide the systems into groups, such that no interfaces exist between the groups and the groups can be performed concurrently. Systems within a single group must be performed sequentially. Such groups are termed management packages because they can be used as a basis for tendering and managing design work.

The CIA can help to explore a diagonal structure of the matrix, and thereby decompose the matrix into mutually separable submatrices. Adapting the CIA originally proposed by Kusiak and Chow (1987), the object–feature matrix was herein changed into an object–object (i.e., successor–predecessor) matrix, because this study is concerned with grouping objects based not on similarities among features but rather on whether the objects exhibit any dependency.

Additionally, finding a separable cluster that includes a system coupled with most of the other systems may not be always possible. Considering such a system when applying the algorithm may yield fewer clusters than would have been obtained were system to have not been considered initially. Some users may wish to remove any such overdependent system before performing the matrix analysis. Therefore, the following terms are defined in this research.

The **cluster dependency threshold** (T) is determined by the user to control the cluster size. The **dependency sum** (D) of a system is the number of other systems that have dependent relationships with the system. D_s of a system is the number of its succeeding systems, and can be determined by summing the numbers in its row in the successor-predecessor matrix. D_p of a system is the number of its preceding systems, and can be determined by summing the numbers of its column in the matrix. D is the maximum of D_s and D_p. Matrix (1) displays D_s and D_p for each of the systems.

The following two sections first describe the proposed version of the CIA algorithm, and then uses the previously described seven systems to demonstrate the algorithm.

Cluster Identification Algorithm

1. Determine the cluster dependency threshold T.
2. Remove from matrix A the systems whose dependency sum D exceeds T, and store these systems in matrix A^*.
3. Set the iteration number, $k=1$.
4. Select any row i of the matrix $A^{(k)}$ (where $A^{(k)}$ denotes matrix A at iteration k) and draw a horizontal line h_i through it.
5. For each entry 1 on the intersection with the horizontal line h_i, draw a vertical line v_j.
6. For each entry crossed by the vertical line v_j, draw a horizontal line h_i.
7. Repeat steps 5 and 6 until no singly crossed entries 1 remain.
8. Transform matrix $A^{(k)}$ into $A^{(k+1)}$ by removing all the twice-crossed entries 1. Add all the twice-crossed entries 1 to A^*.
9. If matrix $A^{(k+1)}=0$ (i.e., all its elements equal zero), stop; otherwise, set $k=k+1$ and go to step 4. Iterations from step 4 to 9 are called the **matrix analysis**.

Each system whose D exceeds T is removed in step 1, and becomes a separated cluster, itself, being uninvolved in the following matrix analysis. A T that is too high may result in too few clusters, such that each cluster is too large (comprising too many systems). Conversely, a T that is too low may result in too many clusters, such that each cluster is too small. Both situations contradict the purpose of using the CIA. The appropriate T depends on the user and the configuration of the matrix. Half of the total number of systems is a recommended starting value for T.

Illustrative Example

Suppose that 3.5 (half of seven systems) is used as the cluster dependency threshold. The following steps show how the CIA processes the example with seven systems [matrix (1)].

1. Set $T=3.5$.
2. S_1 is the only system whose D (Ds) exceeds 3.5. Thus, remove S_1 from A, and store it in A^*. A^* now comprises $\{S_1\}$.
3. Set the iteration number $k=1$.

4–7. Select row S_2 of matrix $A^{(1)}$; draw horizontal line h_2 through it, and then draw a vertical line v_2. Entry 1 at cell (S_2, S_3) is crossed only once. Thus, draw h_3 and then v_3

$$A^{(1)} = \begin{array}{c|cccccc|c} & S_2 & S_3 & S_4 & S_5 & S_6 & S_7 & \\ S_2 & 1 & & & & & & h_2 \\ S_3 & 1 & 1 & & & & & h_3 \\ S_4 & & & 1 & & & & \\ S_5 & & & & 1 & 1 & & \\ S_6 & & & & & 1 & & \\ S_7 & & & & & & 1 & \\ & v_2 & v_3 & & & & & \end{array} \tag{2}$$

8–9. Removing $\{S_2, S_3\}$ from matrix $A^{(1)}$ yields matrix $A^{(2)}$. A^* now comprises $\{S_1\}$ and $\{S_2, S_3\}$.

4–7. Select row S_4 of matrix $A^{(2)}$; draw a horizontal line h_4 through it, and then draw vertical line v_4

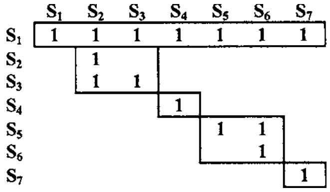


Fig. 1. Clustering result of illustrative example

$$A^{(2)}=\begin{array}{c|cccc|c} & S_4 & S_5 & S_6 & S_7 & \\ S_4 & 1 & & & & h_4\\ S_5 & & 1 & 1 & & \\ S_6 & & & 1 & & \\ S_7 & & & & 1 & \\ & v_4 & & & & \end{array} \tag{3}$$

8–9. Removing $\{S_4\}$ from matrix $A^{(2)}$ yields matrix $A^{(3)}$. A^* now comprises $\{S_1\}$, $\{S_2,S_3\}$ and $\{S_4\}$.
4–7. Select row S_5 of matrix $A^{(3)}$, draw horizontal line h_5 through it, and then draw vertical line v_5 and v_6. Entry 1 at cell (S_6,S_6) is crossed only once. Thus, draw h_6

$$A^{(3)}=\begin{array}{c|ccc|c} & S_5 & S_6 & S_7 & \\ S_5 & 1 & 1 & & h_5\\ S_6 & & 1 & & h_6\\ S_7 & & & 1 & \\ & v_5 & v_6 & & \end{array} \tag{4}$$

8–9. Remove $\{S_5,S_6\}$ from matrix $A^{(3)}$. A^* now comprises $\{S_1\}$, $\{S_2,S_3\}$, $\{S_4\}$, and $\{S_5,S_6\}$.
4–7. Select row S_7 of matrix $A^{(4)}$; draw horizontal line h_7 through it, and then draw vertical line v_7

$$A^{(4)}=\begin{array}{c|c|c} & S_7 & \\ S_7 & 1 & h_7\\ & v_7 & \end{array} \tag{5}$$

8–9. Remove $\{S_7\}$ from matrix $A^{(4)}$. A^* now comprises $\{S_1\}$, $\{S_2,S_3\}$, $\{S_4\}$, $\{S_5,S_6\}$, and $\{S_7\}$. Since $A^{(4)}=0$, stop.

Fig. 1 presents the final clustering result. The system can be decomposed into five clusters, represented by blocks. Except for cluster S_1, all clusters are mutually independent. These clusters are management packages for concurrent design.

Concurrent Design Preplanning

Fig. 2 is a Petri net that shows the concurrent management packages (represented by dotted blocks), based on the decomposed clusters derived in the preceding section. To transform clusters into sequenced activities, activities of different clusters are first separated in different rows. Second, subnetworks showing precedence relationships between activities of each cluster can be reconstructed. Finally, redundant precedence relationships may be removed, and subnetworks integrated as a single network as shown in Fig. 2.

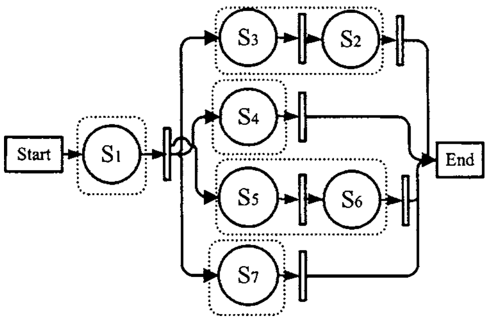


Fig. 2. Example of concurrent design plan

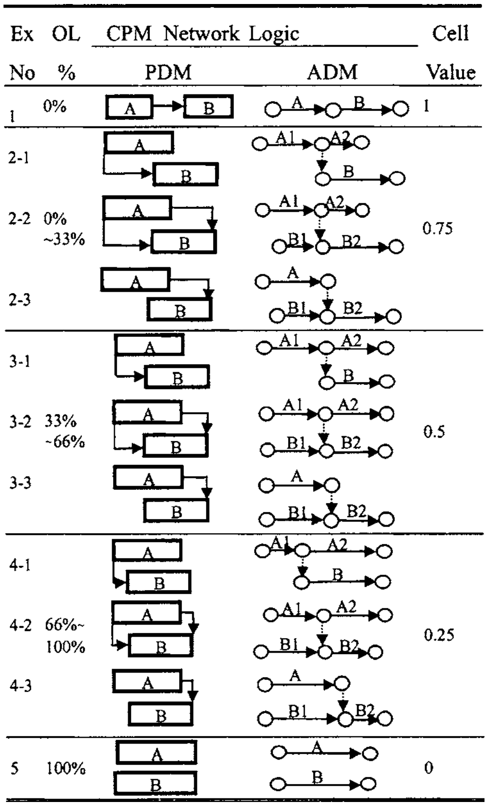


Fig. 3. Measurement of dependency relationships between design activities

Dealing with Dependency Relationships

The previously described matrix contains only binary values. It suffices for designing a manufactured product with distinct parts (e.g., screws and bearings), the designs of which either are independent of, or precede each other with a finish–start relationship. In the design of a wafer FAB, dependencies among systems are more complex, and cannot be neglected because they significantly affect the total design duration. For example, the design of a clean room affects the design of parts of the CSA system (cellular beam and architecture), but not of all of it. Restated, their dependency cannot be characterized as a finish–start relationship.

When a dependency is not binary, it may be described or measured in several ways, such as the percentage overlap of the activities, the network logic (e.g., start–start relationship) as defined in the critical path method (CPM), the amount of dependent information (e.g., the number of drawings or documents), or the required number of coordination meetings. In any case, the dependency must be quantified and dichotomized so the matrix analysis can be performed.

Fig. 3 presents examples of measuring dependency. From top

Table 1. Measurement of Dependency Relationships between Design Activities

	Successor									
System	CSA	MEP	CR	PCW	CH	BG	SG	UPW	WWT	D_s[a]
Predecessor										
CSA	E (1)	M (1)		H (1)	H (1)	H (1)	M (1)	H (1)	H (1)	8
MEP	M (1)	E (1)		L (0)	L (0)	L (0)	L (0)	H (1)	H (1)	8
CR	H (1)	M (1)	E (1)							3
PCW				E (1)						1
CH					E (1)					1
BG						E (1)				1
SG							E (1)			1
UPW								E (1)		1
WWT									E (1)	1
D_p[b]	3	3	1	3	3	3	3	3	3	

Note: Strength of dependency: E (extremely), M (mostly), H (half), L (little), N (not) dependent; Threshold value: M (0.75)

CSA (civil/structure/architecture); MEP (mechanical/electrical/plumbing); CR (clean room); PCW (process cooling water); CH (chemical handling); BG (bulk gas); SG (special gas); UPW (ultra pure water); WWT (waste water treatment)

[a] D_s = the number of the activity's succeeding systems.

[b] D_p = the number of the activity's preceding systems.

row to the bottom, examples 1, 2-1, 2-2, 2-3, 3-1, 3-2, 3-3, 4-1, 4-2, 4-3, and 5 show increasing degree of overlapping between two activities. Column "OL%" measures the dependency as a percentage of overlap of activities. Columns "PDM" and "ADM" show the corresponding time-scaled network schedule, based on the precedence diagramming method (PDM) and the arrow diagramming method (ADM) (Popescu and Charoenngam 1995), respectively. Although PDM is more commonly used in today's scheduling software, such as Primavera Project Planner (Primavera 1999), ADM is also presented because it clarifies which part of an activity precedes which part of a succeeding activity. In addition to the proposed five-point scale of dependency strengths, the three- (Smith and Eppinger 1993) (Austin et al. 1996) and seven-point scales (Rogers and Bloebaum 1994) can also be used.

If an activity can start only when its preceding activity has been completed (Example 1), the two activities will not overlap; thus, their dependency is maximal, and the value in the corresponding cell of the matrix is one. If an activity can start only when a large fraction of the preceding activity has been performed (Example 2-1), then overlap of the activities can be considered to be small; thus, their dependency is strong, and the value in the corresponding cell is 0.75. If an activity can start after a small fraction of its preceding activity has been performed (Example 4-1), then the overlap can be considered to be large; thus, the dependency is weak, and the value in the corresponding cell is 0.25. If the activities are not related (Example 5), then the value in the corresponding cell is zero. The value in a cell represents the strength of a dependency between the corresponding predecessor and successor.

Overlapping activities may also have other types of relationships as described by Examples 2-2, 2-3, 3-2, 3-3, 4-2, and 4-3. When activities have a start–start, both a start–start and a finish–finish, or just a finish–finish relationship, they may overlap. The percentage of the overlap depends on the lead time of the link, determined from the duration of the early part of the preceding activity (e.g., Activity A1 in Column "ADM") and/or the later part of the succeeding activity (e.g., Activity B2).

The matrix analysis requires that the cell values are binary, so the user must determine a threshold (between 0 and 1) to differentiate strong relationships from weak ones. Values above the threshold will be replaced by one, and the others will be replaced by zero. This process is referred to as deoxidization.

Case Study

The case presented here is a 300 mm (12 in.) semiconductor wafer FAB in the Scientific Research Park, Hsinchu, Taiwan. The total construction cost is US$160 million, including the cost of the FAB's structure, mechanics, electrical system, and clean room, but excluding that of the production equipment, which represents alone takes approximately 90% of the total investment in a FAB. The cost of the CSA is US$64 million. The total duration of the project is 13 months, including seven months of design. The project is implemented by the fast-track approach to meet the client's compressed schedule; i.e., the construction starts before the design is completely finished.

The design is contracted out to several specialty engineering firms. All firms have contracts directly with the client. The client is the designated, single window through which design information is exchanged among the firms. The client uses "design start," and 30, 60, 80, and 100% of design completion as milestones to control the schedule. All firms must submit their designs to the client for approval or modification. The client's project management team is responsible for coordinating all the design tasks of the various firms. The management team, led by a chief engineering consultant and consisting mainly of production engineers, has great expertise in designing and operating production lines, but only limited experience of managing a construction project. Consequently, the team may frequently pass information to incorrect parties or fail to pass information that must be passed on, because the team is not familiar with each design area.

This case will be used to illustrate the application of the proposed model.

System Breakdown

The FAB is broken down into nine systems according to the tendering structure of the design tasks, including CSA, mechanical/electrical/plumbing (MEP), clean room (CR), and special sys-

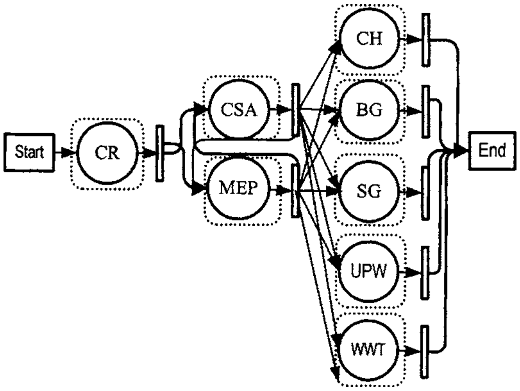


Fig. 4. Initial concurrent design plan for case study

tems. The special systems include process cooling water (PCW), ultrapure water (UPW), waste water treatment (WWT), bulk gas (BG), special gas (SG), and chemical handling (CH) systems. The interfaces among the design firms can be identified and managed by using the design breakdown structure based on the design contracts.

Interface Identification

Two managers from the chief engineering firm in the project help to identify the dependencies among the nine systems, as summarized in Table 1. The dependency is categorized into five degrees-extremely (E), mostly (M), half (H), little (L), and not (N) dependent. For example, CSA is half dependent on CR because the design of CR affects about half of the design work of CSA. For example, the required consideration of microvibrations for CR affects CSA's structural and architectural layout subsystems. MEP is mostly dependent on CR because the design of CR affects more than half of the design work of MEP. For example, the design of CR affects MEP's ice water, product gas, general electricity, HVAC, and fire prevention subsystems.

The binary value in each cell is represented by the numbers in parentheses in the table, using "half dependent" as the threshold. Column "D_s" shows the sum of dependencies of successors for each system. The bottom row "D_p" shows the sum of dependencies of predecessors for each system in the first row. According to the table, CSA and MEP have the highest dependency sums (8).

Management Package Identification

If the cluster dependency threshold T is set to five (half of the nine systems), then CSA and MEP are removed before the matrix analysis is performed. The blocks in Table 1 show the clustering results of the CIA. Fig. 4 represents the concurrent design plan. Notably, CSA depends on MEP and MEP depends on CSA. The cyclical nature of the dependency between CSA and MEP suggests that the two systems should be tendered in a single package, and that they may have to be broken down further.

The further breakdown of a system with a high dependency sum may separate its subsystems with high dependency sums from those with low dependency sums. Such separation isolates the part of the system that is not appropriate for concurrent design, so that the remaining subsystems may be designed concurrently. For example, CSA can be further broken down into four subsystems, including CSA_1 (architectural structure), CSA_2 (architectural layout), CSA_3 (architectural exterior wall), and CSA_4 (architectural landscape). MEP can be further broken down into nine subsystems, including MEP_1 (chiller equipment), MEP_2 (process exhaust system, including general exhaust, acid exhaust, and volatile organic compounds exhaust subsystems), MEP_3 (electrical equipment), MEP_4 (high-voltage equipment), MEP_5 (heating ventilation and air conditioning), MEP_6 (fire protection system), MEP_7 (grounding system), MEP_8 (closed circuit television video equipment), and MEP_9 (plumbing system).

Table 2 displays the strength of dependency relationships among the decomposed subsystems, as assessed by experts, in five degrees as in Table 1. Since from the client's management viewpoint, only significant dependencies need to be captured

Table 2. Professional's Assessment of Dependency among CSA and MEP Systems

	Successor													
Subsystem	CSA_1	CSA_2	CSA_3	CSA_4	MEP_1	MEP_2	MEP_3	MEP_4	MEP_5	MEP_6	MEP_7	MEP_8	MEP_9	D_s[a]
Predecessor														
CSA_1	E(1)	L(0)	—	—	L(0)	—	—	L(0)	—	—	—	—	L(0)	1
CSA_2	M(1)	E(1)	M(1)	E(1)	E(1)	E(1)	E(1)	E(1)	E(1)	E(1)	E(1)	E(1)	E(1)	13
CSA_3	—	—	E(1)	L(0)	—	L(0)	—	—	L(0)	—	—	—	—	1
CSA_4	—	—	—	E(1)	—	—	—	—	—	—	—	—	L(0)	1
MEP_1	E(1)	—	—	—	E(1)	L(0)	L(0)	—	E(1)	—	M(1)	—	—	4
MEP_2	—	—	L(0)	—	—	E(1)	L(0)	—	—	—	M(1)	—	—	2
MEP_3	—	—	—	—	M(1)	M(1)	E(1)	M(1)	M(1)	M(1)	M(1)	M(1)	M(1)	9
MEP_4	—	—	—	—	L(0)	—	L(0)	E(1)	—	—	—	—	—	1
MEP_5	—	—	L(0)	—	E(1)	H(0)	L(0)	—	E(1)	—	L(0)	H(0)	—	2
MEP_6	—	—	—	E(1)	—	—	L(0)	—	—	E(1)	L(0)	—	—	2
MEP_7	—	—	—	—	L(0)	L(0)	E(1)	E(1)	L(0)	L(0)	E(1)	L(0)	L(0)	3
MEP_8	—	—	—	—	L(0)	L(0)	H(0)	H(0)	H(0)	L(0)	—	E(1)	L(0)	1
MEP_9	—	—	—	—	L(0)	L(0)	—	H(0)	H(0)	—	—	—	E(1)	1
D_p[b]	3	1	2	3	4	3	3	4	4	3	5	3	3	

Note: Strength of dependency: E (extremely), M (mostly), H (half), L (little), N (not) dependent; Threshold value: M (0.75)

[a] D_s = the number of the activity's succeeding systems.

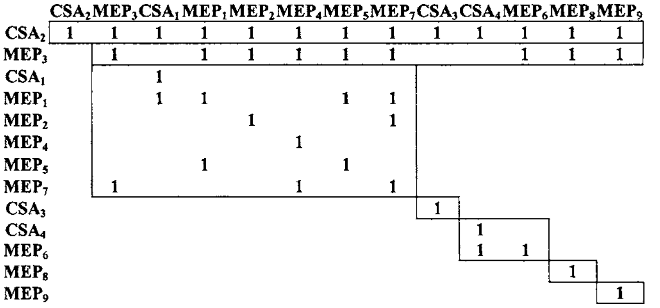
[b] D_p = the number of the activity's preceding systems.

hence avoiding too many coupled subsystems during the matrix analysis, the threshold for deoxidization is set to M. Thus, dependencies identified as E or M will be transformed to 1 and those as H, L, or N will be transformed to 0. For example, although MEP3 is "little" dependent on MEP1 and MEP2 (e.g., the sizing of MEP3 depends on the sizing of MEP1 and MEP2), the dependency will be ignored in the matrix analysis. The deoxidized result is presented by Matrix (6)

$$A = \left(\begin{array}{c|ccccccccccccc}
 & CSA_1 & CSA_2 & CSA_3 & CSA_4 & MEP_1 & MEP_2 & MEP_3 & MEP_4 & MEP_5 & MEP_6 & MEP_7 & MEP_8 & MEP_9 \\
CSA_1 & 1 & & & & & & & & & & & & \\
CSA_2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
CSA_3 & & & 1 & & & & & & & & & & \\
CSA_4 & & & & 1 & & & & & & & & & 1 \\
MEP_1 & 1 & & & & 1 & & & & 1 & & 1 & & \\
MEP_2 & & & & & & 1 & & & & & 1 & & \\
MEP_3 & & & & & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
MEP_4 & & & & & & & & 1 & & & & & \\
MEP_5 & & & & & 1 & & & & 1 & & & & \\
MEP_6 & & & & 1 & & & & & & 1 & & & \\
MEP_7 & & & & & & & 1 & 1 & & & 1 & & \\
MEP_8 & & & & & & & & & & & & 1 & \\
MEP_9 & & & & & & & & & & & & & 1
\end{array}\right) \quad (6)$$

The blocks in Fig. 5 show the clusters derived from the CIA. The decomposed clusters are (CSA_2), (MEP_3), $(CSA_1, MEP_1, MEP_2, MEP_4, MEP_5, MEP_7)$, (CSA_3), (CSA_4, MEP_6), (MEP_8), (MEP_9). In Fig. 6, each dotted block represents a management package that is based on the decomposed clusters. These packages identify groups of tasks that can be performed concurrently, as well as those that cannot. Their dependency relationships among sequential tasks are also identified. For example, suppose that different firms design CSA and MEP. The package (e.g., CSA1 and MEP1) that crosses more than one system should receive considerable management attention because it involves an interface between at least two organizations. The dependency between the two packages within a single system indicates that work interfaces may only exist between different engineers, design teams, or divisions within an organization.

Based on Fig. 4, system CR precedes CSA and MEP, which precede the remaining systems. Matrix (7) displays the dependency relationships between CR and each subsystem of CSA and MEP. Matrix (8) displays the dependency relationships among each subsystem of CSA and MEP, and the other systems. For example, the special consideration of the micro vibration associated with CR governs the design of the box girders in CSA_1. The location of the platform for the CR equipment also governs the



	CSA_2	MEP_3	CSA_1	MEP_1	MEP_2	MEP_4	MEP_5	MEP_7	CSA_3	CSA_4	MEP_6	MEP_8	MEP_9
CSA_2	1	1	1	1	1	1	1	1	1	1	1	1	1
MEP_3		1		1	1	1	1	1			1	1	1
CSA_1			1										
MEP_1			1	1			1	1					
MEP_2					1			1					
MEP_4						1							
MEP_5				1			1						
MEP_7		1				1		1					
CSA_3									1				
CSA_4										1			
MEP_6										1	1		
MEP_8												1	
MEP_9													1

Fig. 5. Clusters derived from cluster identification algorithm

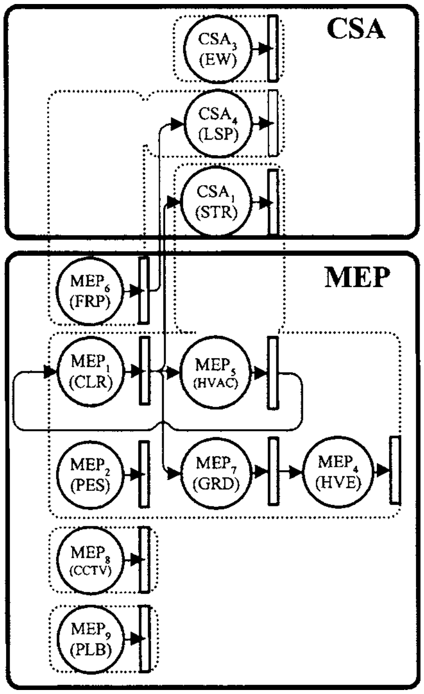


Fig. 6. Concurrent design plan of CSA's and MEP's subsystems

layout of CSA_2. However, the design of PCW requires information on the loading capacity of the slabs and the layout of structural elements and spaces; accordingly, PCW should precede CSA_1 and CSA_2. The dependency relationships among each subsystem and all systems are annotated by bold, dashed lines in Fig. 6. For example, both CSA_2 and MEP_3 precede PCW, CH, BG, SG, UPW, and WWT

$$A = \mathrm{CR}\begin{pmatrix} \mathrm{CSA}_1 & \mathrm{CSA}_2 & \mathrm{CSA}_3 & \mathrm{CSA}_4 & \mathrm{MEP}_1 & \mathrm{MEP}_2 & \mathrm{MEP}_3 & \mathrm{MEP}_4 & \mathrm{MEP}_5 & \mathrm{MEP}_6 & \mathrm{MEP}_7 & \mathrm{MEP}_8 & \mathrm{MEP}_9 \\ 1 & 1 & & & 1 & 1 & 1 & & 1 & 1 & & & \end{pmatrix} \tag{7}$$

$$A = \begin{array}{lcccccc} \mathrm{CSA}_1 & \mathrm{PCW} & \mathrm{CH} & \mathrm{BG} & \mathrm{SG} & \mathrm{UPW} & \mathrm{WWT} \\ \mathrm{CSA}_2 & 1 & & & & 1 & 1 \\ \mathrm{CSA}_3 & 1 & 1 & 1 & 1 & 1 & 1 \\ \mathrm{CSA}_4 & & & 1 & 1 & & \\ \mathrm{MEP}_1 & & 1 & & 1 & & 1 \\ \mathrm{MEP}_2 & 1 & & & & 1 & \\ \mathrm{MEP}_3 & & 1 & & 1 & & \\ \mathrm{MEP}_4 & 1 & 1 & 1 & 1 & 1 & 1 \\ \mathrm{MEP}_5 & & & & & & \\ \mathrm{MEP}_6 & & & & & & \\ \mathrm{MEP}_7 & & & & & 1 & 1 \\ \mathrm{MEP}_8 & & 1 & & 1 & 1 & 1 \\ \mathrm{MEP}_9 & 1 & & & & 1 & 1 \end{array} \tag{8}$$

Resource conflicts may also prevent two processes from running concurrently. The Petri-Net representation enables the plan to be sent to Petri-Net-based simulation tools to verify further the availability of critical resources. For example, some enhanced version of Petri-Net tools (Jensen 1992), (Meta Software Corporation 1993) provide colored tokens to differentiate resources with different attributes and allow dynamic allocation of resources (i.e., a resource may conditionally serve several activities when available).

Discussions

This research was conducted during the design–construction phase of a FAB. Although the analytical results presented above were not actually used to manage the design tasks, the participating design engineers and the client's representatives all appreciated the proposed analytical model and results. They proposed three directions in which the model could be applied; they are in managing design schedule, contract tenders, and information flow.

The concurrent design plan provides a project manager with a reference for estimating quickly the duration of the design phase, and developing a master schedule. For example, suppose that the design of each of the CR, PCW, CH, BG, SG, UPW, and WWT systems requires 30 days, and that the design of each of the CSA1, CSA2, CSA3, CSA4, MEP1, MEP2, MEP3, MEP4, MEP5, MEP6, MEP7, MEP8, and MEP9 subsystems requires 20 days. The longest path associated with the subsystems of CSA and MEP, including four activities, CSA2, CSA3, CSA4 and MEP9 (see Fig. 5), has a duration of 80 days (20×4). Integrating Figs. 4 and 6 yields a total design duration of 120 days (20+80+20).

The project manager may also break down the project into tendering packages, based on the management packages identified in the concurrent design plan. The idea is to combine systems/subsystems that involve the same type of work and have dependency relationships into a single tendering package. Thus, the work interfaces managed by the client in the concurrent design phase can be reduced. For example, MEP1 precedes PCW, and both systems are designed by design firms in the same specialty. Therefore, they can be contracted to a single firm. However, CSA_1 also precedes PCW, but they are designed by firms in different specialties. Even though they are combined into a single tendering package, these tasks will still be subcontracted out by the firm to which the contract is awarded and interfaces still exist between two firms. The difference between tendering the design of such two systems in two separated contracts or a single contract is that the burden of interface management has shifted from the client to the firm to which the pertinent contract has been awarded. The combination of such systems into a single package is not recommended. Nevertheless, the identified interface should still receive considerable attention from the management, regardless of the approach taken.

The concurrent design plan may also support the management of the flow of design information, especially in a situation like that of the studied case, in which all design firms have direct contracts with a client that is unfamiliar with the design process. The arrows in the concurrent design plan indicate the information dependencies. The design activity at an arrow head requires information from the activity at the opposite end of that arrow. Consequently, during the design phase, the client should forward pass the design information of an activity to the firm responsible for the activity's successor when part of the design is completed. When a design change order is initiated for an activity, the client should backward pass the change information to the firm responsible for the activity's predecessor to confirm the feasibility of the change. After the change has been made, the new design should be forward passed to the firm responsible for the activity's successor to realize consequential impacts thereon.

Conclusions

This paper presents an analytical model that supports concurrent design planning. The model involves four steps-system breakdown, system interface identification, management package identification, and concurrent design preplanning. The model supports concurrent design planning by clearly identifying the dependencies among design tasks, and whenever possible, grouping the tasks into distinct management packages, among which is no interface, using the revised CIA. Tasks within a single management package exhibit strong information dependency relationships between each other, and thus are unsuited to be designed concurrently. Tasks in different management packages have weak dependency relationships with each other, and thus can be designed concurrently to shorten the project.

This study also uses a semiconductor wafer FAB to demonstrate the feasibility of applying the proposed model. The participating professionals proposed three directions in which the model can be applied, including the management of design schedule, design contract tendering, and design information flow.

Acknowledgments

The author would like to thank his colleague, Professor W. C. Wang, for his inspiration and input to this research. The following are also appreciated for their participation in our interviews and verification of the case demonstration: C. T. Huang, C. H. Wu, and L. S. Chen (Vice Presidents); Y. P. Hung and W. M. Chen (Directors); S. L. Chiang, C. K. Tang, and S. Y. Yu (Section Managers) at J. J. Pan and Partners, Architects and Planners; and also W. C. Chang (Director), Y. J. Chen, W. C. Tsai, and S. C. Fang (Engineers) at Powerchip Semiconductor Corporation. The writer's graduate student, S. M. Hsieh, is also appreciated for his diligent efforts in coordinating and recording these activities.

References

Ahuja, H. N., Dozzi, S. P., and Abourizk, S. M. (1994). *Project management: Technique in planning and controlling construction projects*, Chaps. 7 and 15, Wiley, New York.

Akao, Y. (1990). *Quality function deployment: Integrating customer requirements into product design*, Productivity Press, Cambridge, Mass.

ALPHATECH, Inc. (1999). "New release—ALPHA/Sim for Windows NT: The essential tool for building high quality, complex models is here." ⟨http://www.alphatech.com/secondary/techpro/alphasim/alphasim.html⟩ (May 20, 2003).

Austin, S., Baldwin, A., and Newton, A. (1996). "A data flow model to plan and manage the building design process." *Mach. Learn.*, 7(1), 3–25.

Baudin, M. (1990). *Manufacturing systems analysis with application to production scheduling*, Prentice-Hall, Englewood Cliffs, N.J.

Black, T. A., Fine, C. F., and Sachs, E. M. (1990). "A method for systems design using precedence relationships: An application to automotive brake systems." *Paper 3208–90*, MIT Sloan School of Management, MIT, Cambridge, Mass.

Boothroyd, G., and Dewhurst, P. (1991). *Product design for assembly*, Boothroyd Dewhurst Inc., Wakefield, R.I.

Browning, T. R. (1998). "Integrative mechanisms for multiteam integration: Findings from five case studies." *J. Syst. Eng.*, 1, 95–112.

Browning, T. R. (2001). "Applying the design structure matrix to system decomposition and integration problems: A review and new directions." *IEEE Trans. Eng. Manage.*, 48(3), 292–306.

Chasey, A. D., and Merchant, S. (2000). "Issues for construction of 300-mm Fab." *J. Constr. Eng. Manage.*, 126(6), 451–457.

Esparza, J., and Lakos, C., eds. (2002). "Application and theory of Petri nets 2002." *Proc., 23rd Int. Conf.: Application and Theory of Petri Nets, ICATPN 2002*, Adelaide, Australia, Springer, New York.

Gordon, F., and Isenhour, R. (1990). "Simultaneous engineering." *Simultaneous engineering integrating manufacturing and design*, Society of Manufacturing Engineering, Dearborn, Mich.

Hoppner, F. (1999). *Fuzzy cluster analysis: Methods for classification, data analysis and image recognition*, Wiley, New York.

ISO/IEC JTC1/SC7 (2004). "Official ISO/IEC JTC1/SC7 web site." ⟨http://www.jtc1-sc7.org/⟩ (July 25, 2004).

Jansen, R. (2000). "The 300 mm FAB—A major paradigm shift." *Semiconductor FABTEC*, IGC Publishers, London, 113–120.

Jensen, K. (1992). *Coloured Petri nets. Basic concepts, analysis methods and practical use*, Vol. 1, Springer, Berlin.

Kusiak, A., and Chow, W. S. (1987). "An efficient cluster identification algorithm." *IEEE Trans. Syst. Man Cybern.*, SMC-17(4), 696–699.

Meta Software Corporation (1993). *Design/CPN*, Meta Software Corporation, Cambridge, Mass.

Peña-Mora, F., and Li, M. (2001). "Dynamic planning and control methodology for design/build fast-track construction projects." *J. Constr. Eng. Manage.*, 127(1), 1–17.

Popescu, C. M., and Charoenngam, C. (1995). *Project planning, scheduling, and control in construction: An encyclopedia of terms and applications*, Wiley, New York.

Primavera Systems, Inc. (1999). *Primavera Project Planner 3.0 reference manual*, Bala Cynwyd, Pa.

Rask, I., and Sunnersjö, S. (1998). "Design structure matrices for the planning of rule-based engineering systems." *Proc., Europe Conf. on Integration in Manufacturing*, Chalmers Univ. of Technology, Göteborg, Sweden, 158–165.

Reisig, W. (1992). *A primer in Petri net design*, Springer, Berlin.

Rogers, J. L., and Bloebaum, C. L. (1994). "Ordering design tasks based on coupling strengths." *Paper 94–4326*, American Institute of Aeronautics and Astronautics, Reston, Va.

Semiconductor Industry Association (SIA). (2003). *National technology roadmap for semiconductor technology needs*, SEMATECH, Inc., Austin, Tex.

Smith, R. P., and Eppinger, S. D. (1993). "Characteristics and models of iteration in engineering design." *Proc., Int. Conf. on Engineering Design* (ICED'93), The Hague, The Netherlands, 564–571.

Steward, D. V. (1981). "The design structure system: A method for managing the design of complex systems." *IEEE Trans. Eng. Manage.*, 28, 71–74.

Suh, N. P. (1990). *The principle of design*, Oxford University Press, New York.

Suh, N. P. (1995). "Axiomatic design of mechanical systems." *Trans. Am. Soc. Civ. Eng.*, 117, 2–10.

Turtle, Q. (1994). *Implementing concurrent project management*, Prentice–Hall, Englewood Cliffs, N.J.