Grouping of Tasks for Cooperative Product Development

Takeshi Murayama[#1], Toru Eguchi[*], Akira Ogawa[**]
[#] Dept. of Medical System and Biomaterial Engineering, Hiroshima University
1-2-3 Kasumi, Minami-ku, Hiroshima 734-8553, Japan
[1]murayatk@hiroshima-u.ac.jp
[*] Dept. of Mechanical Systems Engineering, Hiroshima University
1-4-1 Kagamiyama, Higashi-hiroshima 739-8527, Japan
[**] Kobe Steel Ltd.

Abstract— A lot of members from various disciplines are involved in the development of complex products, and this brings about large-scale, complicated cooperation among them. Such cooperation causes the following problems:

- **the time consumption for achieving a consensus among them increases; and**
- **the management of product development becomes difficult.**

In order to develop the products efficiently, this paper proposes a method for dividing the product development activity into several work groups on an appropriate scale, in each of which some of the members can cooperate with each other without the problems mentioned above.

In this method, the product development activity is represented by a directed graph, each of which nodes expresses a task and each of which arcs expresses the flow of data/information between the tasks. ISM (Interpretive Structural Modeling) identifies strongly connected sub-graphs, in each of which the members should cooperate with each other. However, if a strongly connected sub-graph is large, the problems mentioned above still exist. In this case, we use Bottleneck Method to divide the sub-graph into several smaller sub-graphs, each of which corresponds to a work group. The effectiveness of the method is demonstrated by showing an example of constructing the work groups.

Keyword- Product Development, Cooperative Design, Concurrent Engineering, Interpretive Structural Modeling, Graph theory

I. Introduction

Concurrent engineering is a work methodology based on the parallelization of tasks (i.e. performing tasks concurrently). It refers to an approach used in product development in which functions of design and manufacturing, and other functions are integrated to reduce the elapsed time required to bring a new product to the market[1,2]. Cooperative product development[3,4], in which members from various disciplines (e.g., mechanical, electronic, and software design, manufacturing, marketing, and so on) are involved, is one of the important fields in the concurrent engineering.

To develop complex products, a lot of members are involved, and this brings about large-scale, complicated cooperation among them. Such cooperation causes the following problems:

- the time consumption for achieving a consensus among them increases; and
- the management of product development becomes difficult.

In order to develop the products efficiently, this paper proposes a method for dividing the product development activity into several work groups on an appropriate scale, in each of which some of the members cooperate with each other without the problems mentioned above. ISM (Interpretive Structural Modeling) [5] and Bottleneck Method[6] are used for constructing several work groups on an appropriate scale. As the related works, matrix-based methods have been developed[7,8,9,10], which can sequence and/or group tasks. As compared with the related works, our method provides a way to divide a large strongly connected work group into smaller work groups on an appropriate scale.

II. Methods

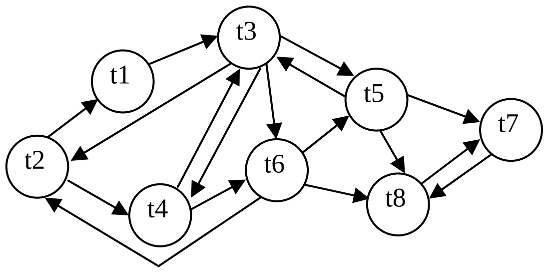
A. Representation of Product Development Activity

In this method, a product development activity is represented by a directed graph, each of which nodes expresses a task and each of which arcs expresses the flow of data/information between the tasks. We refer to the graph as "task network." The task network is also represented by an adjacency matrix, *TN*. In this matrix, the entry tn_{ij} , which denotes the value of the *i*th row and the *j*th column, is:

1, if the data/information are passed from task t_i to task t_j; and

0, otherwise.

Figure 1 shows an example of the task network and adjacency matrix.



(a) Task network

	t1	t2	t3	t4	t5	t6	t7	t8
t1	0	0	1	0	0	0	0	0
t2	1	0	0	1	0	0	0	0
t3	0	1	0	1	1	1	0	0
t4	0	0	1	0	0	1	0	0
t5	0	0	1	0	0	0	1	1
t6	0	1	0	0	1	0	0	1
t7	0	0	0	0	0	0	0	1
t8	0	0	0	0	0	0	1	0

(b) Adjacency matrix

Fig. 1 Task network and adjacency matrix

B. Application of ISM

ISM[5] enables us to develop a map of the complex relationships among many elements, by decomposing the complex system into several subsystems.

We apply ISM to the task network, in order to identify several strongly connected sub-graphs. Each strongly connected sub-graph expresses a work group in which members should cooperate with each other.

In the ISM procedure, we need to find the value of *K* that satisfies Eq. (1), to identify the strongly connected sub-graphs.

$$(I+TN)^K = (I+TN)^{K+1} \quad (1)$$

where *I* is the unit matrix and the calculation is based on Boolean Algebra.

$(I+TN)^K$ is called the reachability matrix, and we express this matrix as *M*. Namely,

$$M = (I+TN)^K \quad (2)$$

Figure 2 shows the reachability matrix for the task network shown in Fig.1.

	t1	t2	t3	t4	t5	t6	t7	t8
t1	1	1	1	1	1	1	1	1
t2	1	1	1	1	1	1	1	1
t3	1	1	1	1	1	1	1	1
t4	1	1	1	1	1	1	1	1
t5	1	1	1	1	1	1	1	1
t6	1	1	1	1	1	1	1	1
t7	0	0	0	0	0	0	0	1
t8	0	0	0	0	0	0	1	0

Fig. 2 Reachability matrix

Next, for each task, t_i, the reachability set, R_i, and the antecedent set, A_i, are found, by Eqs. (3) and (4), respectively.

$$R_i = \left\{ t_j \in T \middle| m_{ij} = 1 \right\} \quad (3)$$

$$A_i = \left\{ t_j \in T \middle| m_{ji} = 1 \right\} \quad (4)$$

where T is a set of tasks and m_{ij} denotes the value of the ith row and the jth column in M. The reachability set, R_i, is the set of tasks reachable from the task t_i in the task network. The antecedent set, A_i, is the set of tasks that are reachable to the task t_i.

Then we can find a strongly connected sub-graph that satisfies Eq. (5).

$$R_i \cap A_i = R_i \quad (5)$$

This is the set of the tasks that can be reached to each other and can not be reached to the other tasks that are not included in the set.

Next we remove the tasks mentioned above from M, and we find the next strongly connected sub-graph by Eqs.(3), (4), and (5). The same procedure is repeated until no strongly connected sub-graph is found. Figure 3 shows the strongly connected sub-graphs for the task network shown in Fig.1.

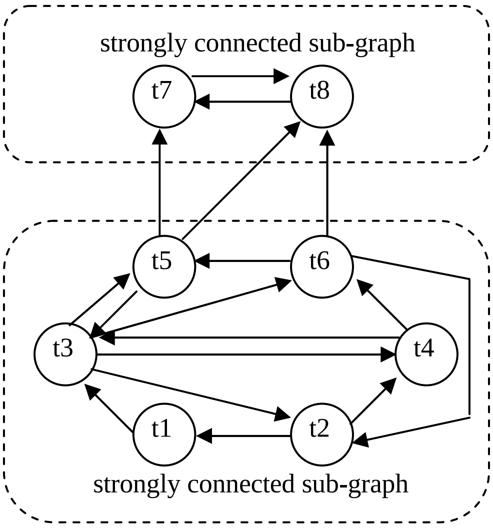


Fig.3 Strongly connected sub-graphs identified by ISM

C. Application of Bottleneck Method

As we mentioned, each of the strongly connected sub-graphs expresses a work group in which members should cooperate with each other. However, if a strongly connected sub-graph is large, the problems mentioned in the introduction still exist. In this case, we use Bottleneck Method[6] to divide the strongly connected sub-graph into several smaller sub-graphs.

According to Bottleneck Method, we first calculate all of the combinations of the tasks included in a strongly connected sub-graph which we want to divide. Next we calculate the shortest path for each combination (t_i, t_j). Then for each arc we calculate the number that the shortest paths pass through. Figure 4 shows an example, in which the numbers are shown to the arcs. Table I shows a shortest path for each combination (t_i, t_j), and Table II shows the number that the shortest paths pass through an arc. The arcs having big numbers are considered as bottle necks in the strongly connected sub-graph. There is a great possibility that the sub-graph is divided by cutting such arcs. We cut such arcs until the strongly connected sub-graph is divided into smaller sub-graphs. In the example shown in Fig.4, Table I and II, we can divide the strongly connected graph into two strongly connected graphs by cutting the arcs, t2->t5 and/or t7->t4 which have the biggest number, 16.

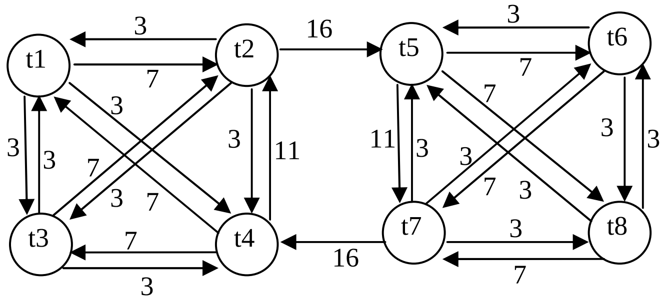


Fig. 4 Task network having the number that the shortest paths pass through an arc

TABLE I
Shortest Path for Each Combination of the Tasks. t_i->t_j Means an Arc from t_i to t_j.

Combination of tasks	Shortest path	Combination of tasks	Shortest path
(t1, t2)	t1->t2	(t5, t1)	t5->t7->t4->t1
(t1, t3)	t1->t3	(t5, t2)	t5->t7->t4->t2
(t1, t4)	t1->t4	(t5, t3)	t5->t7->t4->t3
(t1, t5)	t1->t2->t5	(t5, t4)	t5->t7->t4
(t1, t6)	t1->t2->t5->t6	(t5, t6)	t5->t6
(t1, t7)	t1->t2->t5->t7	(t5, t7)	t5->t7
(t1, t8)	t1->t2->t5->t8	(t5, t8)	t5->t8
(t2, t1)	t2->t1	(t6, t1)	t6->t7->t4->t1
(t2, t3)	t2->t3	(t6, t2)	t6->t7->t4->t2
(t2, t4)	t2->t4	(t6, t3)	t6->t7->t4->t3
(t2, t5)	t2->t5	(t6, t4)	t6->t7->t4
(t2, t6)	t2->t5->t6	(t6, t5)	t6->t5
(t2, t7)	t2->t5->t7	(t6, t7)	t6->t7
(t2, t8)	t2->t5->t8	(t6, t8)	t6->t8
(t3, t1)	t3->t1	(t7, t1)	t7->t4->t1
(t3, t2)	t3->t2	(t7, t2)	t7->t4->t2
(t3, t4)	t3->t4	(t7, t3)	t7->t4->t3
(t3, t5)	t3->t2->t5	(t7, t4)	t7->t4
(t3, t6)	t3->t2->t5->t6	(t7, t5)	t7->t5
(t3, t7)	t3->t2->t5->t7	(t7, t6)	t7->t6
(t3, t8)	t3->t2->t5->t8	(t7, t8)	t7->t8
(t4, t1)	t4->t1	(t8, t1)	t8->t7->t4->t1
(t4, t2)	t4->t2	(t8, t2)	t8->t7->t4->t2
(t4, t3)	t4->t3	(t8, t3)	t8->t7->t4->t3
(t4, t5)	t4->t2->t5	(t8, t4)	t8->t7->t4
(t4, t6)	t4->t2->t5->t6	(t8, t5)	t8->t5
(t4, t7)	t4->t2->t5->t7	(t8, t6)	t8->t6
(t4, t8)	t4->t2->t5->t8	(t8, t7)	t8->t7

Table II
The Number that the Shortest Paths Pass Through an Arc

Arc	Number that the shortest paths pass through an arc, t_i->t_j	Arc	Number that the shortest paths pass through an arc, t_i->t_j
t1->t2	7	t5->t6	7
t1->t3	3	t5->t7	7
t1->t4	3	t5->t8	11
t2->t1	3	t6->t5	3
t2->t3	3	t6->t7	7
t2->t4	3	t6->t8	3
t2->t5	16	t7->t4	16
t3->t1	3	t7->t5	3
t3->t2	7	t7->t6	3
t3->t4	3	t7->t8	3
t4->t1	7	t8->t5	3
t4->t2	11	t8->t6	3
t4->t3	7	t8->t7	7

However this method may make us cut an important arc (i.e., an important relation between two tasks). Therefore we give the importance degree to each arc and choose the arcs to be cut, referring not only the number mentioned above but also the importance degree. In this study, we give the importance degree to each arc, as follows:

$$\text{Importance degree} = \begin{cases} 1\,; \text{the relation between two tasks is less important} \\ 2\,; \text{the relation between two tasks is important} \\ 3\,; \text{the relation between two tasks is very important} \end{cases} \qquad (6)$$

III. CALCULATION EXAMPLE

Using the data described in [7], we constructed the cooperative work groups by the proposed method. Figure 5 shows the task network that we drew according to the data and Table III shows the types of tasks included in the task network. Figure 6 shows the result of applying ISM to the task network. As shown in this figure, a large group is constructed, in which most of the tasks are included. As this group is too large, we applied Bottleneck Method to this group. Table IV shows the number that the shortest paths pass through an arc. In addition the importance degree of each arc is also given, as shown in Table IV. An operator chose and cut some arcs, referring the number and importance degree shown in Table IV. As a result, we obtained four work groups shown in Fig. 7. In this figure, the dotted arcs mean ones cut by the operator. The result could be different, depending on how the operator chooses the arcs to be cut.

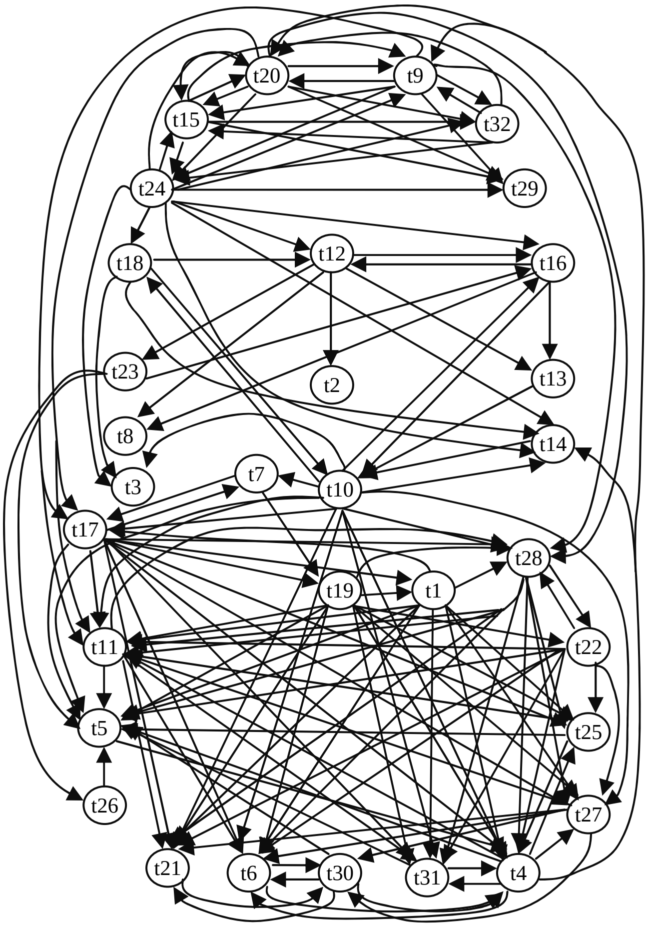


Fig. 5 Task network

Table III
Types of Tasks Included in the Task Network

Task number	Task Type	Task number	Task Type
tl	Determine FMEA for Process	t17	Develop Process Concept
t2	Discuss with PFT	t18	Coordinate Design
t3	Develop Mockup & Illustration	t19	Determine Key Quality Char.
t4	Reso1ve Problems	t20	Develop Design Concept
t5	Verify Design	t21	Outsources Analysis
t6	In-House Analysis	t22	Determine FMEA for System
t7	Analyze and Build / Test Results	t23	Check Geometry Mockup
t8	Write Detail Ticket	t24	Determine Physical Requirement
t9	Develop Process Concept	t25	Analyze Manufacturability
t10	Store and Distribute Drawings	t26	Analyze Structure
t11	Develop Quality Plan	t27	Coordinate Analysis
t12	Schedule Work	t28	Develop Equipment Concept
t13	Write Specification	t29	Initiate/Monitor Build/Test
t14	Detail Check Camshaft	t30	Verify Analysis Results
t15	Develop Production Concept	t31	Analyze Tolerance
t16	Create Layout	t32	Determine Primary Characteristics

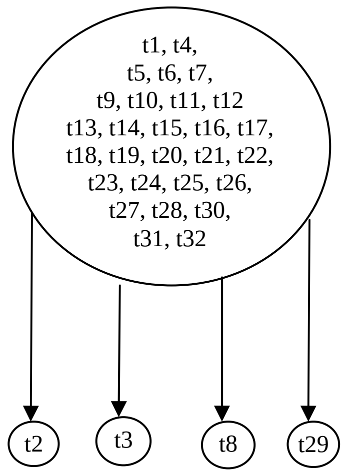


Fig. 6 Result of applying ISM to the task network

Table IV
The Number that the Shortest Paths Pass Through an Arc, and the Importance Degree of Each Arc

Arc	Number	Importance degree	Arc	Number	Importance degree	Arc	Number	Importance degree	Arc	Number	Importance degree
tl->t4	10	3	t10->t16	30	3	t17->t19	31	3	t22->t31	1	3
tl->t5	1	3	t10->t17	32	3	t17->t21	16	3	t23->t16	43	3
tl->t6	2	3	t10->t18	20	3	t17->t22	33	3	t23->t26	35	3
tl->t11	1	3	t10->t21	22	3	t17->t25	11	3	t24->t9	27	3
tl->tl7	6	3	t10->t25	11	3	t17->t27	22	3	t24->t12	59	2
tl->t21	2	3	t10->t27	22	3	t17->t28	4	3	t24->t14	24	3
tl->t25	1	3	t10->t28	16	2	t17->t31	11	3	t24->t15	25	3
tl->t27	2	3	t10->t31	11	3	t18->t10	47	2	t24->t16	57	3
tl->t28	2	3	t11->t4	17	3	t18->t12	14	2	t24->t18	32	2
tl->t31	1	3	t11->t5	10	3	t18->t14	1	2	t24->t20	27	3
t4->t5	4	1	t11->t6	10	3	t19->t1	2	3	t24->t32	3	3
t4->t9	81	1	t11->t21	10	3	t19->t4	24	3	t25->t4	1	2
t4->t11	22	2	t11->t25	10	3	t19->t5	2	3	t25->t5	19	2
t4->t14	47	2	t11->t27	14	3	t19->t6	4	3	t26->t5	74	2
t4->t15	42	1	t11->t28	14	3	t19->t11	2	3	t27->t6	64	2
t4->t20	148	1	t11->t31	10	3	t19->t17	5	3	t27->t21	5	2
t4->t25	H	3	t12->t13	67	2	t19->t21	4	3	t27->t30	102	2
t4->t27	18	3	t12->t16	44	2	t19->t25	2	3	t28->t4	15	3
t4->t31	H	3	t12->t23	36	2	t19->t27	4	3	t28->t5	10	3
t5->t4	52	3	t13->t10	62	3	t19->t28	3	3	t28->t6	8	3
t6->t4	41	3	t14->t10	81	3	t19->t31	2	3	t28->t11	3	3
t6->t30	39	3	t15->t9	19	3	t20->t9	3	3	t28->t17	14	3
t7->t17	52	3	t15->t11	24	2	t20->t11	21	2	t28->t21	8	3
t7->t19	57	3	t15->t17	47	2	t20->t15	3	3	t28->t22	36	3
t9->t11	21	2	t15->t20	19	3	t20->t17	65	2	t28->t25	10	3
t9->t15	3	3	t15->t24	19	2	t20->t24	87	2	t28-	14	3

t9->t17	44	2	t15->t28	33	2	t20->t28	39	2	t28->t31	10	3
t9->t20	3	3	t15->t32	19	2	t20->t32	19	2	t30->t4	65	3
t9->t24	41	2	t16->t10	93	3	t21->t30	102	3	t30->t5	4	2
t9->t28	30	2	t16->t12	12	3	t22->t4	10	3	t30->t6	4	3
t9->t32	19	2	t16->t13	30	3	t22->t5	1	3	t30->t21	4	3
t10->t4	49	3	t17->t1	31	3	t22->t6	2	3	t31->t4	19	3
t10->t5	10	3	t17->t4	29	3	t22->t11	1	3	t31->t5	1	2
t10->t6	22	3	t17->t5	11	3	t22->t21	2	3	t32->t9	27	3
t10->t7	20	3	t17->t6	16	3	t22->t25	1	3	t32->t15	25	3
t10->t11	8	3	t17->t7	24	3	t22->t27	2	3	t32->t20	27	3
t10->t14	8	3	t17->t11	2	3	t22->t28	7	3	t32->t24	13	2

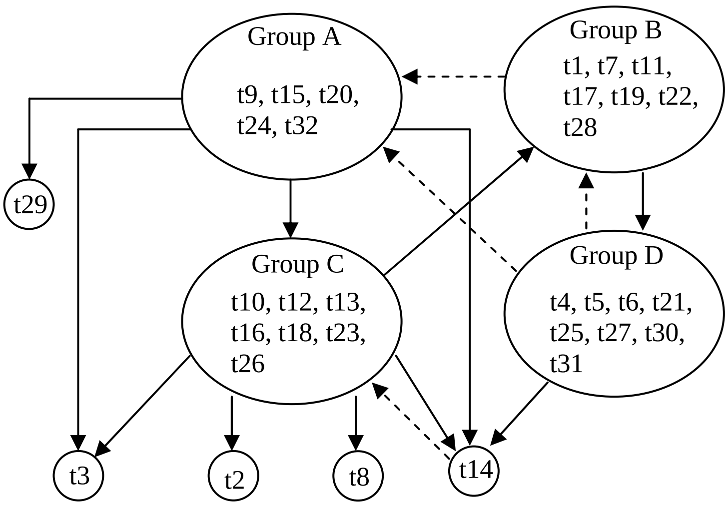


Fig. 7 Constructed work groups by applying Bottleneck Method

IV. CONCLUSION

We proposed the method of constructing the cooperative work groups by ISM and Bottleneck Method.

The calculation example showed that the method can be used for constructing the cooperative work groups on an appropriate scale even if a large strongly connected sub-graph (i.e., a large work group) is included in the task network. Our further work will be to develop a way of allocating members to each of the work groups identified by the proposed method.

REFERENCES

[1] (2015) Wikipedia, "Concurrent engineering." Available: http://en.wikipedia.org/wiki/ Concurrent_ engineering

[2] R. A. Tenkorang, "Concurrent Engineering (CE): A Review Literature Report," in Proceedings of the World Congress on Engineering and Computer Sciences, San Francisco. 2011.

[3] D. Sriram, R. Logcher, S. Fukuda, Computer-Aided Cooperative Product Development, Springer-Verlag, 1991.

[4] T. Murayama, H. Fukumaru, H. Matsuoka, F. Oba, “Sharing and adjustment of design targets in a distributed CAD system,” in Proceedings of 2002 IEEE International Conference on Industrial Technology, Vol.1, pp.138 – 142, 2002.

[5] J. N. Warfield, "Toward Interpretation of Complex Structural Model," IEEE Transactions on Systems, Man & Cybernetics, Vol.4, No.5, pp.405-417, 1974.

[6] T. Terano and N. H. Qvan, “Decomposition of Strongly Connected Graphs by Bottle Neck Method and Its Applications,” Transactions of the Society of Instrument and Control Engineers, Vol.12, No.6, pp.681-686, 1976.

[7] S. D. Eppinger et al., “Organizing the Tasks in Complex Design Projects,” Computer-Aided Cooperative Product Development, Springer Verlag, 1991.

[8] D. V. Steward, “The Design Structure System: A Method for Managing the Design of Complex Systems,” IEEE Trans. Engineering Management, Vol.28, No.3, pp. 71-74,1981.

[9] A. Kusiak et al., “Concurrent Engineering: Decomposition and Scheduling of Design Activities,” Int. J. Production Research, Vol.28, No.10, pp.1883-1900 ,1990.

[10] (2015) Structure Matrix (DSM). Available: http://www.dsmweb.org/

AUTHOR PROFILE

Takeshi Murayama is a professor at Hiroshima University since 2005. He received his master degree from University of Osaka Prefecture in 1986, and his doctor degree from Hiroshima University in 1993. His current interests include CAD/CAM, manufacturing systems, medical systems and application of 3D technologies (e.g., 3D modeling, 3D printing and 3D scanning).