Automatic Detection of Latent Bottleneck Processes through Perturbation-based Repercussion Analysis

Seiichi Komiya, Daisuke Kinoshita, Hiroki Uchikawa, and Yuya Kosaka

***Abstract*—**Typical organizations in a company are usually engaged in more than one project running in parallel. In such a case, human and non-human resources (i.e., workers and development environments) are inevitably shared by those projects to carry out respective tasks. When a resource is shared among two or more projects and if a change is requested for such a resource by one of the projects, there may be a situation in which the project cannot change its assigned resource or adjust its schedule since other processes are competing for the same resource at the same time. The authors call such a process a bottleneck process. If a delay spreads to one of the bottleneck processes, the project cannot be completed by its due date because it is not possible to adjust its schedule or to change the resource. It is necessary to take measures to prevent a process delay from spreading to bottleneck processes. This paper proposes a method to detect bottleneck processes automatically and discusses the effectiveness of the method by simulating the impacts (perturbation) generated by delays of preceding processes using Perturbation-based Repercussion Analysis.

***Keywords*—**Project management, Detection of bottleneck processes, Perturbation, Repercussion analysis

I. Introduction

Development of large-scale software is usually conducted by a project to enable joint cooperation of all workforces. Regardless to the life cycle model adopted, a development plan is an essential factor of a software development project, including task scheduling for development and assignment of personnel to each task. Therefore, to lead a project to a successful conclusion, it is necessary to establish management objectives based on the software development plan and to check their attainment levels.

Typical organizations in a company usually carry out more than one project in parallel. In such a case, human and non-human resources (i.e., workers and development environments) are inevitably shared by those projects to carry out respective tasks. When a resource is shared among two or more projects and if a change is requested for such a resource by one of the projects, there may be a situation in which the project cannot change its assigned resource or adjust its schedule since other processes are competing for the same resource at the same time. The authors call such a process a bottleneck process. If a delay spreads to such a process, the project cannot be completed by its due date because it is not possible to adjust its schedule or to change the resource. It is necessary to take measures to prevent a process delay from spreading to bottleneck processes. However, it is not easy to detect a potential bottleneck process in advance because it is necessary to recognize not only both the development plans of own project and other projects but also their schedules changing day by day.

The authors propose a method to detect bottleneck processes automatically and discuss the effectiveness of the method by simulating the impacts (perturbation) generated by delays of preceding processes using Perturbation-based Repercussion Analysis.

This paper contains the following sections. Section 2 discusses the type and the nature of constraints inherent to the planning issues of software development. Section 3 introduces related works. Section 4 describes the definitions of bottleneck process and potential bottleneck process as well as their examples. Section 5 introduces an analysis method that can be used for repercussion analysis of impacts on succeeding processes generated by delays of preceding processes. Section 6 describes several case studies to which our analysis method is applied based on the examples shown in Section 4. Section 7 discusses the effectiveness of the proposed analysis method based on the consideration of applied case studies (repercussion analysis of impacts generated by delays of preceding processes) shown in Section 6. Section 8 describes the conclusion.

II. Constraints Inherent to Software Development Planning

In this research, the authors recognize the conditions that a software development plan has to satisfy as constraints [1, 2]. The constraints that are inherent to planning of software development are as follows:

(1) Constraints imposed by the execution sequence of work

In a software development project, the execution sequence of processes is determined by their intermediate software products. For example, as for Process b in Figure 1,

Process b is feasible only after the intermediate software products α has been produced from Process a. This is called the pre-condition of Process b. The intermediate software product β must have been produced from Process b before Process c is started. This is called the post-condition of Process b. Thus the execution sequence of Processes a, b, and c is determined by the

intermediate software products α and β. Such a condition is called the constraint on the execution sequence of tasks.

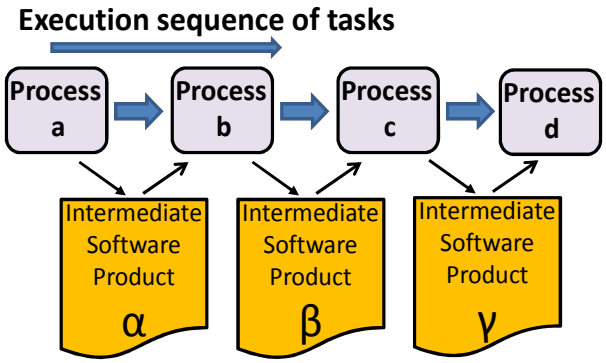

Fig. 1 constraints on the execution sequence of tasks based on intermediate software products

(2) Constraints imposed by competent resource allocation

Only skilled, qualified and competent human resources (i.e., workforce) and non-human resources (such as machine environments) with satisfactory level of feasibility can be assigned to each task of software development. Such a constraint is called the constraint imposed by competent resource allocation. For example, the processes related to a specific programming language, system testing, or debugging can be performed only by persons who are able to do such tasks. Therefore, the schedule of software development tasks varies depending on the conditions of human and non-human resources, in other words, the constraints on resource assignment.

(3) Constraints imposed by available periods of resources

Even if there are competent resources for software development tasks, those resources cannot be assigned to a specific task when they are engaged in other work. Resources can be assigned only when they are available. Such a constraint is called the constraint imposed by available period of resource allocation.

(4) Constraints imposed by resources' limitations

We introduce the concept of capacity as an attribute of each resource to express resource capability limitations. A resource's capacity is defined by the upper limit value of each resource's working rate (in percentage). The working rate is obtained by dividing the total working hours of a resource per day (when one resource is simultaneously assigned to several tasks, the total working hours are calculated as the sum of those working hours) by the available working hours of that resource and then multiplying by 100. The working rate upper limit value calculated in advance is defined as the constraints imposed by the resource capability limitation. For example, suppose that Worker Pl is assigned to both Work A (two days workload) and Work B (two days workload) for a week (five days), the working rate of Worker P1 for the week is 80%. In this case, if the working rate upper limit of Worker P1 is set to no less than 80%, it is possible to assign him/her as described above. However, if the value is less than 80%, it is not possible to assign him/her. Thus the working rate can be used as a scale for evaluating a worker's workload and for checking if the worker is overloaded. This concept can be applied to non-human resources. Typical capacities (working rate upper limit) may vary depending on the rank of resources. By default, the working rate upper limit is set to 100% or eight hours.

If the combination of resources assigned to each process of a project satisfies all the constraints, it can be considered as a candidate of software development plan. That is, planning a software development project schedule can be considered as solving an optimization problem of combinations with many constraints.

III. Related Work

Various models have been proposed so far to represent the task structure of a software development project, including PMDB [7], Design-Net [8, 9], Kyoto DB [10], and PROMX [11]. However, these models do not explicitly address either the relationship between software development tasks and the resources essential to conduct the tasks or the constraints on the conditions and available periods of resource assignment although they are useful for the models to represent the task model of a project because they focus on how to represent the hierarchy and sequence of tasks. (Although PMDB uses Person as an entity, it does not address the constraints related to the resource allocation conditions and the available period of resources.) Therefore, they are not adequate for the project management models of software development projects.

Finally we mention CCPM (Critical Chain Project Management) [6, 13, 16, 17] getting attention recently in comparison with our approach. We have to explain TOC (Theory of Constraints) [13, 14, 15] before discussing CCPM. TOC is a management method that focuses on the weakest portion (Constraint Conditions in TOC terms) in company activities and reinforces and improves that portion intensively to achieve a maximum success with minimum efforts. Based on the idea of TOC, CCPM, as a project management method, performs optimization from the viewpoint of entire project. CCPM uses Critical Chain in place of conventional Critical Path and removes the additional part of efforts included in the estimation phase for safety purpose to shorten each process period (specifically, adopts an effort estimation of 50% success probability), and then add a project buffer (margin days) at the end of a process on a Critical Path to manage the entire process at a single point. In addition, it inserts a joining buffer between the tasks on the Critical Chain path and the tasks on the path that joins to the Critical Chain to prevent the Critical Chain from being affected by delays of tasks on the path that joins to the Critical Chain path. Then the plan of project schedule is created by taking into account the delivery date, cost and constraints of the resources. Then the project manager understands the progress of the whole project by examining the consumption ratio of the buffer instead of managing the progress of each process. The above is an outline of CCPM.

Our approach is different from CCPM in the following points:

- Our viewpoint of man-hours estimation is different from that of CCPM. CCPM adopts man-hours estimation of each process with 50% of success

probability and uses the joining and project buffers to reduce the risk of process delay due to estimation errors. In our approach, an average of extra man-hours is calculated for the joining and project buffers and assigned to each process.

- Our viewpoint of progress management is different from that of CCPM. CCPM manages the progress of the whole project by examining the consumption ratio of the buffer instead of managing the progress of each process. For this reason, CCPM can be used to detect process delays of the whole project, but it is not adequate for understanding the progress of processes which are not on the Critical Chain. In our approach, progress is managed by each process. As a result, it is possible to understand the progress of every process, regardless of if it is on the Critical Path.

When a process delay is detected, it is not easy to change the project schedule in CCPM to recover the delay, but in our approach as described in [4, 5], it is possible to use our tool to develop a revised plan dynamically that can be used to recover the process delay.

IV. Definitions of Bottleneck Process and Potential Bottleneck Process and Their Case Studies

(1) Definition of bottleneck process and its case study

Suppose that a single resource is being used in more than one project. There is a situation in which the schedule of a project cannot be adjusted or the resource used in the process cannot be replaced when a delay spread to the process. Such a process is called a bottleneck process. Figure 2 shows a case of such a situation.

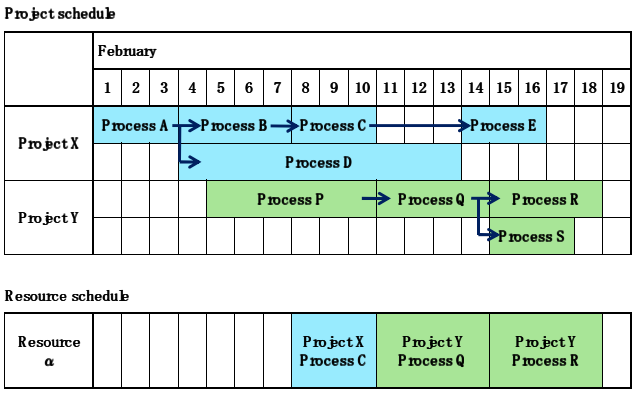

Fig. 2 case study for bottleneck processes

The project schedule is located at the top of Figure 2 and the arrows indicate the "constrains on the sequence of tasks." Specifically, the upper part of Figure 2 represents a constraint on the sequence of tasks which indicates that the tasks of Processes B and D are started only after the tasks of Process A have been completed. The lower part of Figure 2 represents the schedule of resource usage. For convenience, suppose that only Resource α can be assigned to Process C. Figure 2 shows a situation in which Resource α is assigned to Process C of Project X and Processes Q and R of Project Y. Note that there are no margin days between the three consecutive Processes C, Q, and R to which Resource α is assigned.

In this case, if a delay occurs in Process B of Project X, the start and end dates of Process C drop behind the schedule for the delayed days. When examining only the schedule of Project X, it would appear that the schedule can be adjustable for a maximum of three days to deal with the delay by setting back the start and end dates of Process C for the delayed days since there is three margin days for Process C. However, adjusting the schedule in this way prevents Resource α from being assigned to Process Q because Resource α is sequentially assigned to Processes C, Q, and R and setting back the schedule results in a delay of the start date of Process Q and prevents Process Q from keeping three working days, as indicated by the constraints on the available periods of resource assignment. Therefore the project manager has to wait until 19 before he/she can assign Resource α to Process C, after which Resource α is available for consecutive three days. As a result, the end date of the project will be significantly delayed leading the project to a failure unless Process Q of Project Y is reinforces by adding personnel or replacing resources. For this reason, the task schedule of Process C cannot be moved at all. In addition, the resource to be used in Process C cannot be replaced since only Resource α can be assigned to Process C. Based on the above discussion, Process C is determined to be a bottleneck process.

To lead a project to a success, it is essential to prevent any delay from spreading to a bottleneck process even if it is a minimum of one day, while conducting the project. To keep the schedule of a bottleneck process intact, any delay should be cleared when the process (Process B in Figure 2) just before the bottleneck process (Process C in Figure 2) has completed.

(2) Definition of potential bottleneck process and its case study

There is a situation in which a process which is not an original bottleneck process changes into a bottleneck process due to a delay of its start date caused by a delay of a preceding process. Such a process is called a potential bottleneck process.

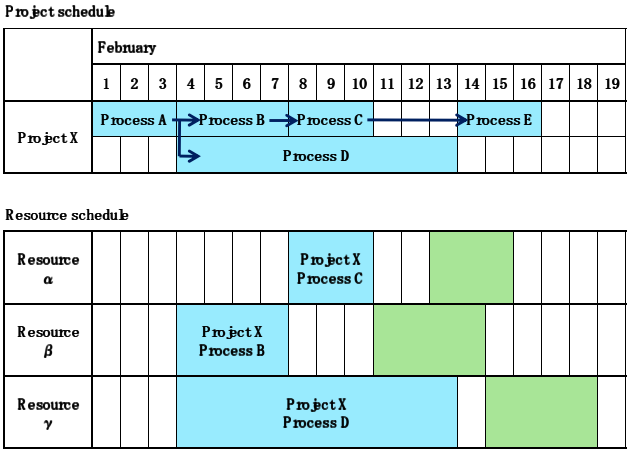

Fig. 3 case study 2 for bottleneck processes

The following describes potential bottleneck processes according to Figure 3. Suppose that Resources β, α, and γ are assigned to Processes B, C, and D respectively and there is no assignable resource other than β, α, and γ. In addition, suppose that Processes A and E assume no constraint and can be delayed any days.

In Figure 3, only Resources α is assignable to Process C. In addition, the allowable delay of Process C is two days due to the constraint of Resource α. For this reason, Process C cannot adjust its task schedule or replace its resources when a process delay of two days spreads to Process C. That is, a two days process delay changes Process C into a bottleneck process.

There is no resource other than Resource β that can be assigned to Process B. In addition, it seems that Process B can be delayed for a maximum of three days according to the constraint of Resource β. However, a two days process delay makes Process B unable to adjust its task schedule or replace its resource because a two days process delay changes succeeding Process C into a bottleneck process. That is, a two days process delay changes Process B into a bottleneck process.

Process A has two succeeding Processes B and D. Since a two days process delay changes Process B that follows Process A into a bottleneck process, it is determined that only a delay of two days is allowed for the path including Processes A, B, C, and E in sequence. On the other hand, examining the path including Process A and D makes it clear that Process A can never be a bottleneck process since its resource has no constraint. Whereas, only Resource γ is assignable to Process D. In addition, Process D can be delayed for only one day according to the constraint of Resource γ. As a result, a one day process delay makes Process D unable to adjust its task schedule or replace its resource. That is, a one day process delay changes Process D into a bottleneck process. It can be determined that only a one day process delay is allowed for Process A that is preceding to Process D in the path including Processes A and D. Comparing the both processes lead to a conclusion that only a one day process delay is allowed for Process A.

Based on the above discussion about the case study of Figure 3, there are three potential bottleneck Processes B, C, and D, which change into actual bottleneck processes by two days, two days, and one day process delays, respectively.

V. Our Method of Perturbation-based Repercussion Analysis on Process Delay

In this paper, the authors propose a method of perturbation-based repercussion analysis on process delay which simulates the impacts on succeeding processes which are generated by various process delays that are assumed in preceding processes.

Figure 4 illustrates the procedure of perturbation-based repercussion analysis. Numbers are corresponding to the descriptions that follow.

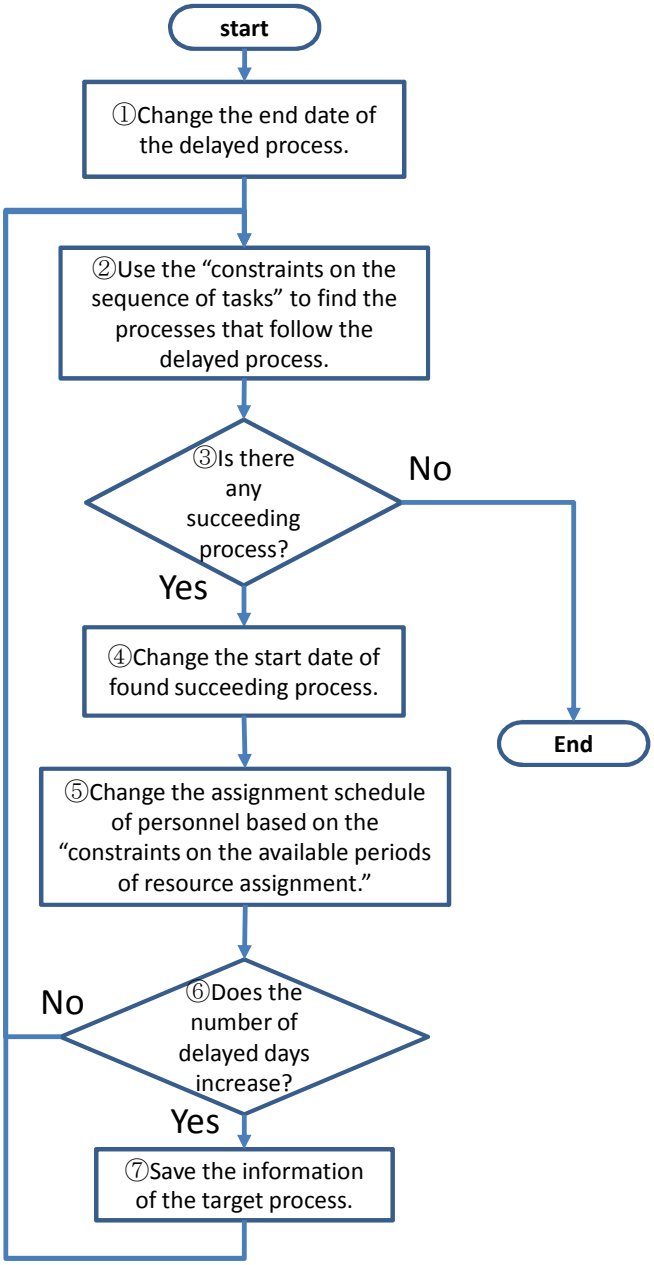

Fig. 4 The flow of perturbation-based repercussion analysis

- Change the end date of the delayed process.
- Use the "constraints on the sequence of tasks" to find the processes that follow the delayed process.
- Based on the search results, check to see if there is a succeeding process.
- Change the start dates of all the succeeding processes that require changes to their start dates.
- According to the changed start dates, change the assignment schedule of personnel based on the "constraints on the available periods of resource assignment." However, the schedules of projects that are not analysis target must not be changed. If the person is assigned to another project during the target period of changes, assign the person after the other schedule is finished.
- Check to see if the number of delayed days increases

after the person is reassigned.

- Save the information of the process of which delay expanded since it is highly possible that the process schedule requires adjustment.

Repeat the procedure from (2) to (7) and terminate the perturbation-based repercussion analysis when no succeeding process is found or the project is completed.

VI. Results of Simulation Performed Using the Proposed Analysis Method

The following are simulation results produced by applying the analysis method (simulation method) to the case study illustrated in Figure 3.

(1) When the first delay arises in Process A:

Supposing that the first delay arises in Process A, change the schedule of Process A by adding an assumed number of delayed days. The number of days added as perturbation (delay) is increased by day, starting with one day in the first simulation, two days in the second simulation, and five days (one week) in the last simulation. The authors set the maximum perturbation days as one week since the progress management of software development project is usually performed once in a week.

(i) When a one day delay is assumed in Process A:

In this case, the end date of Process A is delayed for one day and falls on February 4 since there is no constrains on the available periods of assignment. Duration of execution periods of Processes B and D that follow Process A are also delayed for one day since the end date of Process A is delayed for one day. Since a one day delays of Processes B and D do not violate the constraint on the assignment period of Resource β assigned to Process B and D, the execution periods of Processes B and D are set to the periods from February 5 to 8 and from February 4 to 14, respectively.

Since the end date of Process B is delayed for one day, the execution period of Process C that follows Process B is set to the period from February 9 to 11 at the earliest. Since a one day delay of Processes C does not violate the constraint on the assignment period of Resource α assigned to Process C, the execution period of Processes C is set to the period from February 9 to 11.

Although Process E that follows both Processes C and D is affected by the delays of Processes C and D, the execution period of Process E is delayed for only one day since there is no constraint on the available period of assignment of resource assigned to Process E. Thus the execution period of Process E is set to the period from February 15 to 17.

(ii) When a two days delay is assumed in Process A:

In this case, the same approach used in the case in which a one day delay is assumed in Process A can be used to determine the start and end dates of each process. As a result, the end date of Process A is delayed for two days and falls on February 5. The execution period of Process B is set to the period from February 6 to 9, changed by the impact of the delay of Process A. The execution period of Process C is set to the period from February 10 to 12, changed by the impact of the delay of Process B.

The execution period of Process D is set to the period from February 6 to 15 at the earliest, changed by the impact of the delay of Process A. However, Resource γ assigned to Process D has already been assigned to another project from February 15 to 18, so it is unavailable in this period. In addition, since there is no resource other than Resource γ which can be assigned to Process D, the execution period of Process D should be set to the period from February 19 to 28 during which Resource γ is available. Thus, at this simulation stage it is revealed that Process D has fallen into a potential bottleneck process at the previous stage, that is, in the simulation case of a one day delay.

As for the impact of the delay of Process C, the execution period of Process E stays unchanged from February 14 to 16. However, is changed to the period from March 1 to 3 at the earliest according to the delay of Process D. Here, since there is no constraint on the assignment period of resource assigned to Process E, the execution period of Process E is determined to be set to the period from May 1 to 3.

(iii) When a three days delay is assumed in Process A:

In this case, the same approach used in the case in which a one day delay is assumed in Process A can be used to determine the start and end dates of each process. As a result, the end date of Process A is delayed for three days and falls on February 6. The execution period of Process B is set to the period from February 7 to 10, changed by the impact of the delay of Process A. The execution period of Process C is set to the period from February 11 to 13 at the earliest, changed by the impact of the delay of Process B. However, since Resource α assigned to Process C is scheduled to be assigned to another project in the period from February 13 to 15, it cannot be used in this period. In addition, since there is no resource other than Resource α which can be assigned to Process C, the execution period of Process C should be set to the period from February 16 to 18 during which Resource α is available. Thus, at this simulation stage it is revealed that Process C has fallen into a potential bottleneck process at the previous stage, that is, in the simulation case of a two days delay.

The execution period of Process D is set to the period from February 7 to 16 at the earliest, changed by the impact of the delay of Process A. However, since Resource γ assigned to Process D is scheduled to be assigned to another project in the period from February 15 to 18, it cannot be used in this period. In addition, since there is no resource other than Resource γ which can be assigned to Process D, the execution period of Process D should be set to the period from February 19 to 28 during which Resource γ is available.

The execution period of Process E is set to the period from February 19 to 21, changed by the impact of the delay of Process A and it is set to the period from March 1 to 3 at the earliest, changed by the impact of the delay of Process D. Here, since there is no constraint on the assignment period of resource assigned to Process E, the execution period of Process E can be determined to be set to the period from May 1 to 3.

(iv) When a four days delay is assumed in Process A:

In this case, the same approach used in the case in which a one day delay is assumed in Process A can be used to determine the start and end dates of each process. As a result, the end date of Process A is delayed for four days and falls on February 7. The

execution period of Process B is set to the period from February 8 to 11, changed by the impact of the delay of Process A. However, since Resource β assigned to Process B is scheduled to be assigned to another project in the period from February 11 to 14, it cannot be used in this period. In addition, since there is no resource other than Resource β which can be assigned to Process B, the execution period of Process B should be set to the period from February 15 to 18 during which Resource β is available. Thus, at this simulation stage it is revealed that Process B has fallen into a potential bottleneck process at the previous stage, that is, in the simulation case of a three days delay. The execution period of Process C is set to the period from February 19 to 21, changed by the impact of the delay of Process B.

The execution period of Process D is set to the period from February 8 to 17 at the earliest, changed by the impact of the delay of Process A. However, since Resource γ assigned to Process D is scheduled to be assigned to another project in the period from February 15 to 18, it cannot be used in this period. In addition, since there is no resource other than Resource γ which can be assigned to Process D, the execution period of Process D should be set to the period from February 19 to 28 during which Resource γ is available.

The execution period of Process E is set to the period from February 22 to 24, changed by the impact of the delay of Process A and it is set to the period from March 1 to 3 at the earliest, changed by the impact of the delay of Process D. Here, since there is no constraint on the assignment period of resource assigned to Process E, the execution period of Process E can be determined to be set to the period from May 1 to 3.

(v) When a five days delay is assumed in Process A:

In this case, the same approach used in the case in which a one day delay is assumed in Process A can be used to determine the start and end dates of each process. As a result, the end date of Process A is delayed for five days and falls on February 8. The execution period of Process B is set to the period from February 9 to 12, changed by the impact of the delay of Process A. However, since Resource β assigned to Process B is scheduled to be assigned to another project in the period from February 11 to 14, it cannot be used in this period. In addition, since there is no resource other than Resource β which can be assigned to Process B, the execution period of Process B should be set to the period from February 15 to 18 during which Resource β is available. The execution period of Process C is set to the period from February 19 to 21, changed by the impact of the delay of Process B.

The execution period of Process D is set to the period from February 9 to 18 at the earliest, changed by the impact of the delay of Process A. However, since Resource γ assigned to Process D is scheduled to be assigned to another project in the period from February 15 to 18, it cannot be used in this period. In addition, since there is no resource other than Resource γ which can be assigned to Process D, the execution period of Process D should be set to the period from February 19 to 28 during which Resource γ is available.

The execution period of Process E is set to the period from February 22 to 24, changed by the impact of the delay of Process A and it is set to the period from March 1 to 3 at the earliest, changed by the impact of the delay of Process D. Here, since there is no constraint on the assignment period of resource assigned to Process E, the execution period of Process E can be determined to be set to the period from May 1 to 3.

(2) When the first delay arises in Process B:

In this case, the execution period of Process D that follows Process A is set to the period from February 4 to 13 as initially scheduled since there is no delay in Process A.

(i) When a one day delay is assumed in Process B:

In this case, the end date of Process B is delayed for one day and falls on February 8. The execution period of Process C that follows Process B is also delayed for one day and set to the period from February 9 to 11. The execution period of Process E is set to the period from February 14 to 16 as initially scheduled.

(ii) When a two days delay is assumed in Process B:

In this case, the end date of Process B is delayed for two days and falls on February 9. The execution period of Process C that follows Process B is also delayed for two days and set to the period from February 10 to 12. The execution period of Process E is set to the period from February 14 to 16 as initially scheduled.

(iii) When a three days delay is assumed in Process B:

In this case, the end date of Process B is delayed for three days and falls on February 10. Thus the execution period of Process C is set to the period from February 11 to 13. However, since Resource α assigned to Process C is scheduled to be assigned to another project in the period from February 13 to 15, the execution period of Process C should be set to the period from February 16 to 18 during which Resource α is available. Thus, at this simulation stage it is revealed that Process C has fallen into a potential bottleneck process at the previous stage, that is, in the simulation case of a two days delay.

Since there is no delay in Processes A and D, the execution period of Process E is set to the period from February 19 to 21 at the earliest, changed only by the impact of the delay of Process C.

(iv) When a four days delay is assumed in Process B:

In this case, the end date of Process B is delayed for four days and falls on February 11. However, since Resource β assigned to Process B is scheduled to be assigned to another project in the period from February 11 to 14, the end date of Process B falls on February 15 on which Resource α is available. Thus, at this simulation stage it is revealed that Process B has fallen into a potential bottleneck process at the previous stage, that is, in the simulation case of a three days delay. The execution period of Process C is set to the period from February 16 to 18, changed by the impact of the delay of Process B.

Since there is no delay in Processes A and D, the execution period of Process E is set to the period from February 19 to 21 at the earliest, changed only by the impact of the delay of Process C.

(v) When a five days delay is assumed in Process B:

In this case, the end date of Process B is delayed for five days and falls on February 12. However, since Resource β assigned to Process B is scheduled to be assigned to another project in the period from February 11 to 14, the end date of Process B falls on

February 16 on which Resource α is available. The execution period of Process C is set to the period from February 17 to 19, changed by the impact of the delay of Process B.

Since there is no delay in Processes A and D, the execution period of Process E is set to the period from February 20 to 22 at the earliest, changed only by the impact of the delay of Process C.

(3) When the first delay arises in Process C:

Since there is no delay in Processes A and B, the execution period of Process B is set to the period from February 4 to 7 as initially scheduled, and the execution period of Process D that follows Process A is set to the period from February 4 to 13 as initially scheduled.

(i) When a delay of one or two days is assumed in Process C:

In these cases, the end date of Process C falls on February 11 and 12 given a one day and two days delay, respectively. However, since there are three additional days available for Process C, Process E that follow Process C is not affected by these delays and the execution period of Process E is set to the period from February 14 to 16 as initially scheduled.

(ii) When a delay of three, four or five days is assumed in Process C:

In these cases, the end date of Process C falls on February 13, 14, and 15 given a three, four, and five days delay, respectively. However, since Resource α assigned to Process C is scheduled to be assigned to another project in the period from February 13 to 15, the end date of Process C falls on February 16, 17 and 18, respectively, on which Resource α is available. Thus, at this simulation stage it is revealed that Process C has fallen into a potential bottleneck process at the previous stage, that is, in the simulation case of a two days delay.

Since there is no delay in Processes A, B, and D, the execution period of Process E is set to the period from February 17 to 19, from 18 to 20, and from 19 to 21, given a three days, four days, and five days delays, respectively.

(4) When the first delay arises in Process D:

In this case, Processes A, B, and C are executed as scheduled initially since there is no delay in these processes.

(i) When a one day delay is assumed in Process D:

The end date of Process D is delayed for one day and falls on February 14.

The execution period of Process E that follows Process D is set to the period from February 15 to 17, changed by the impact of the delay of Process D.

(ii) When a delay from two to five days is assumed in Process D:

In these cases, the end date of Process D falls on February 15, 16, 17, and 18 given a two, three, four, and five days delay, respectively. However, since Resource γ assigned to Process D is scheduled to be assigned to another project in the period from February 15 to 18, the end date of Process D falls on February 19, 20, 21, and 22, respectively, on which Resource γ is available. Thus, at this simulation stage it is revealed that Process D has fallen into a potential bottleneck process at the previous stage, that is, in the simulation case of a one day delay.

Since there is no constraint on the assignment period of Process E that follows Process D, the execution period of Process E is set to the period from February 20 to 22, from 21 to 23, from 22 to 24, and from 23 to 25 given a two, three, four, and five days delays, respectively.

(5) When the first delay arises in Process E:

In this case, Processes A, B, C and D are executed as scheduled initially since there is no delay in these processes.

Since there is no constraint on the resources assigned to Process E, the impact of a delay in Process E remains in just the same number of days as the initial delay. That is, the number of delayed days is not amplified by the constraints on the assignment periods.

VII. Discussion on Effectiveness Of The Proposed Method Based on the Case Studies

In Section 6, the authors described the method of the perturbation-based repercussion analysis, which can be used to analyze the impacts on the succeeding processes generated by delays in preceding processes using the case studies illustrated in Figure 3. Table 1 shows the analysis results.

We used the method of the perturbation-based repercussion analysis to figure out the resultant number of delayed days by assuming various numbers of initial delayed days (perturbation). Table 1 shows the results.

(1) The following lists the cases in which the number of resultant perturbation days is greater than the number of the initial perturbation days. However, delayed processes that simply inherit the delays of the preceding processes are excluded.

- A two days delay of Process A results in a fifteen days delay of Process D.
- A three days delay of Process A results in an eight days delay of Process C.
- A four days delay of Process A results in a eleven days delay of Process B.
- A three days delay of Process B results in an eight days delay of Process C.
- A four days delay of Process B results in an eight days delay of Process B.
- A three days delay of Process C results in a six days delay of Process C.
- A two days delay of Process D results in a six days delay of Process D.

(2) As for a process for which the number of resultant perturbation days is greater than the number of the assumed perturbation days, it is revealed that such a process has fallen into a bottleneck process when a one day smaller number of perturbation days is specified. The following lists such cases.

- Process D falls into a bottleneck process when a one day delay is assumed in Process A.
- Process C falls into a bottleneck process when a two days delay is assumed in Process A.
- Process B falls into a bottleneck process when a three days delay is assumed in Process A.
- Process C falls into a bottleneck process when a two days delay

is assumed in Process B.
- Process B falls into a bottleneck process when a three days delay is assumed in Process B.
- Process C falls into a bottleneck process when a two days delay is assumed in Process C.
- Process D falls into a bottleneck process when a one day delay is assumed in Process D.

In this paper, we assumed that the perturbation unit is one day, but the unit other than one day can also be used to detect a bottleneck process. For example, assuming the perturbation unit as two days also generates the results (1) and (2).

The analysis method proposed in this paper can be used to check which process falls in a bottleneck process and how many days of delay makes it a bottleneck process, in other word, it can be used to examine the details of potential bottleneck processes. If the details of potential bottleneck processes are known before they actually arise as bottleneck processes, they can be addressed in advance with proactive countermeasures to prevent a delay from spreading to such potential bottleneck processes.

When an actual delay is detected and suspected to spread to a potential bottleneck process, a countermeasure must be developed to resolve the process delay before it spread to a potential bottleneck process. The following three countermeasures can be adopted:

(i) Apply a crashing operation to the process just before the bottleneck process.
(ii) Apply a crashing operation to the bottleneck process.
(iii) Change the resource allocation to prevent the resources used by the bottleneck process from being used in the development plans of other projects.

Crashing indicates that allocating a large volume of resources available. The authors have implemented a tool that can automatically generate a plan to recover process delays based on (i) and (ii) when a process delay is detected, and published a paper [2].

On the other hand, as for (iii), a fully automatic countermeasure cannot be developed since human intervention is indispensable when selecting a top-priority project for crashing from a number of competing projects.

In this paper, we used a simplified version of case studies for convenience. For example, since nonworking days such as Saturday, Sunday, or national holidays are not taken into account, a delay of five days is actually a delay of a week and can be felt longer.

In addition, in the case studies used in this paper, a single resource is assigned to a number of projects to clarify the effectiveness of our method, although in a practical project, such an assignment is impractical. The authors consider that the case studies are useful to show the effectiveness of our method because the essential point of the problem is not changed.

VIII. Conclusion

The method proposed in this paper enables detection of bottleneck processes using the perturbation-based repercussion analysis that simulates how various delays assumed spread to the processes in a project. In addition, it also enables to develop a proactive countermeasure by using the perturbation-based repercussion analysis before a bottleneck arises, which is traditionally used after an actual bottleneck has arisen.

In this paper, the authors discussed the bottleneck processes that arose in a project and their problems. We also discussed the method to detect bottleneck processes automatically using the perturbation-based repercussion analysis and its effectiveness. We also clarified that applying a crashing operation to the process before a bottleneck process detected can prevent delays from spreading to the bottleneck process.

References

[1] Seiichi Komiya, Naota Sawabe, Atsuo Hazeyama, “Constraint-Based Schedule Planning for Software Development,” The Transactions of IEICE D-I Vol.J79-D-I No.9, pp.544-557, 1996.
[2] S. Komiya, A. Hazeyama, “A Meta-Model of Work Structure of Software Project and a Framework for Software Project Management System,” IEICE TranINF SYST, vol.E81-D.No12, pp1415-1428, Dec 1998
[3] A. Hazeyama, S. Komiya, “Workload Management Facilities for Software Project Management,” IEICE Tran INF SYST, vol.E81-D.No12 , pp1404-1414, Dec 1998.
[4] R. Yaegashi, D. Kinoshita, H. Hashiura, K. Uenosono, S. Komiya, "Automatically Creating a Schedule Plan as Countermeasure by Means of "Crashing" against Process Delay," JCKBSE'04, pp.24-36, Protvino, Russia, Aug. 2004.
[5] Rihito Yaegashi, Daisuke Kinoshita, Hiroaki Hashiura, Kazuhiro Uenosono, Yuuichiro Hyashi, Seiichi Komiya, “Automatically Creating a Fast-Tracking-Based Countermeasure Plan against Process Delay,” The Transactions of IEICE D-I, Vol. J88-D-I, No.2, pp.215-227, 2005.
[6] Leach, Lawrence P., “Critical Chain Project Management”, Artech House Professional Development Library, London 313, 2000
[7] M. H. Penedo and E. D. Stuckle, “PMDB - A project master database for software engineering environments,” 8th International Conference on Software Engineering, pp.150-157, 1985.
[8] L. Liu and E. Horowitz, “A formal model for software project management,” IEEE Trans. on Software Engineering, vol.15, no.10, pp.1280-1293, Oct. 1989.
[9] L .Liu and E. Horowitz, “Object database support for a software project management environment,” ACM SIGSOFT Software Engineering Notes, vol.13, no.5, pp.85-96, Nov.1988.
[10] Y. Matsumoto and T. Ajisaka, “A data model in the software project database KyotoDB,” JSSST Advances in Software Science and Technology 2, pp103-121, 1990.
[11] H. Sato, ”Project management expert system,” Proc.ACM CSC’87, Feb.1987.
[12] D. Kinoshita, R. Yaegashi, H. Hashiura, K. Uenosono, S. Komiya, “An Automatic Schedule Planning System: Strategies and Evaluation for Implementing the System,” Joint Conference on Knowledge-Based Software Engineering 2004 (JCKBSE’04), pp.37-48, Protvino, Russia, Aug.2004.
[13] Kimio Inagaki “TOC Critical Chain Revolution”, JMA Management Center Inc., 1998.
[14] Eliyahu M. Goldratt, “The Goal”, DIAMOND, Inc., 2001.
[15] Eliyahu M. Goldratt, “The Goal 2”, DIAMOND, Inc., 2002.
[16] Eliyahu M. Goldratt, “Critical Chain”, DIAMOND, Inc., 2003.

[17] Hidetaka Nakajima, Koji Tsumagari, " PROJECT MANAGEMENT CRITICAL CHAIN," JMA Management Center Inc., 2003.
[18] S. Komiya A. Hazeyama, "A Meta-Model of Work Structure of Software Project and a Framework for Software Project Management System, "IEICE Tran INF SYST , vol. E81-D. No12, pp1415-1428, Dec 1998
[19] Daisuke Kinoshita, Rihito Yaegashi, Kazuhiro Uenosono, Hiroaki Hashiura, Hiroki Uchikawa, and Seiichi Komiya, "Automatic Creation of a Crashing-Based Schedule Plan as Countermeasures against Process Delay"INTERNATIONAL JOURNAL of SYSTEMS APPLICATIONS, ENGINEERING & DEVELOPMENTIssue 4, Volume 2, 2008, pp. 170-177.

Seiichi Komiya He received a bachelor's degree from Saitama University, Japan in 1969.He received Doctorate in engineering from Shinshu University, Japan, in 2000. At present, he is Professor at Shibaura Institute of Technology from 2001.
Daisuke Kinoshita He received a bachelor's degree from Shibaura Institute of Technology, Japan in 2002. He received a master's degree from Graduate School of Engineering, Shibaura Institute of Technology, Japan, in 2004. He received doctorate in engineering from Graduate School of Engineering, Shibaura Institute of Technology, Japan, in 2010. At present, he is working in Hitachi, Ltd.
Hiroki Uchikawa He received a bachelor's degree from Shibaura Institute of Technology, Japan in 2007. He received a master's degree from Graduate School of Engineering, Shibaura Institute of Technology, Japan, in 2009. At present, he is working in Hitachi, Ltd.
Yuya Kosaka He received a bachelor's degree from Shibaura Institute of Technology, Japan in 2009. And he takes a master's course, Graduate School of Engineering, Shibaura Institute of Technology.

TABLE I
SIMULATION RESULTS OF THE PERTURBATION-BASED REPERCUSSION ANALYSIS

Assumed perturbation days		Process A	Process B	Process C	Process D	Process E
Process A Three days	One day	Feb. 1 – 4 one day delay	Feb. 5 – 8 one day delay	Feb. 9 – 11 one day delay	Feb. 5 – 14 one day delay (Bottleneck)	Feb. 15 – 17 one day delay
	Two days	Feb. 1 – 5 two days delay	Feb. 6 – 9 two days delay	Feb. 10 – 12 two days delay (Bottleneck)	Feb. 19 – 28 fifteen days delay	Mar. 1 – 3 fifteen days delay
	Three days	Feb. 1 – 6 three days delay	Feb. 7 – 10 three days delay (Bottleneck)	Feb. 16 – 18 eight days delay	Feb. 19 – 28 fifteen days delay	Mar. 1 – 3 fifteen days delay
	Four days	Feb. 1 – 7 four days delay	Feb. 15 – 18 eleven days delay	Feb. 19 – 21 eleven days delay	Feb. 19 – 28 fifteen days delay	Mar. 1 – 3 fifteen days delay
	Five days	Feb. 1 – 8 five days delay	Feb. 15 – 18 eleven days delay	Feb. 19 – 21 eleven days delay	Feb. 19 – 28 fifteen days delay	Mar. 1 – 3 fifteen days delay
Process B Four days	One day	Feb. 1 – 3 no delay	Feb. 4 – 8 one day delay	Feb. 9 – 11 one day delay	Feb. 4 – 13 no delay	Feb. 14 – 16 no delay
	Two days	Feb. 1 – 3 no delay	Feb. 4 – 9 two days delay	Feb. 10 – 12 two days delay (Bottleneck)	Feb. 4 – 13 no delay	Feb. 14 – 16 no delay
	Three days	Feb. 1 – 3 no delay	Feb. 4 – 10 three days delay (Bottleneck)	Feb. 16 – 18 eight days delay	Feb. 4 – 13 no delay	Feb. 19 – 21 five days delay
	Four days	Feb. 1 – 3 no delay	Feb. 4 – 15 eight days delay	Feb. 16 – 18 eight days delay	Feb. 4 – 13 no delay	Feb. 19 – 21 five days delay
	Five days	Feb. 1 – 3 no delay	Feb. 4 – 16 nine days delay	Feb. 17 – 19 nine days delay	Feb. 4 – 13 no delay	Feb. 20 – 22 six days delay
Process C Three days	One day	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 11 one day delay	Feb. 4 – 13 no delay	Feb. 14 – 16 no delay
	Two days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 12 two days delay (Bottleneck)	Feb. 4 – 13 no delay	Feb. 14 – 16 no delay
	Three days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 16 six days delay	Feb. 4 – 13 no delay	Feb. 17 – 19 three days delay
	Four days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 17 seven days delay	Feb. 4 – 13 no delay	Feb. 18 – 20 four days delay
	Five days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 18 eight days delay	Feb. 4 – 13 no delay	Feb. 19 – 21 five days delay
Process D Ten days	One day	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 14 one day delay (Bottleneck)	Feb. 15 – 17 one day delay
	Two days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 19 six days delay	Feb. 20 – 22 six days delay
	Three days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 20 seven days delay	Feb. 21 – 23 seven days delay
	Four days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 21 eight days delay	Feb. 22 – 24 eight days delay
	Five days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 22 nine days delay	Feb. 23 – 25 nine days delay
Process E Three days	One day	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 13 no delay	Feb. 14 – 17 one day delay
	Two days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 13 no delay	Feb. 14 – 18 two days delay
	Three days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 13 no delay	Feb. 14 – 19 three days delay
	Four days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 13 no delay	Feb. 14 – 20 four days delay
	Five days	Feb. 1 – 3 no delay	Feb. 4 – 7 no delay	Feb. 8 – 10 no delay	Feb. 4 – 13 no delay	Feb. 14 – 21 five days delay