

DESCRIPTION

Title of Invention: System development assistance device and system development assistance method

Technical field

[0001]

The present invention relates to a system development supporting device and a system development supporting method, and more particularly, to a technique capable of coping with defects from a design stage in an upstream process and supporting rapid and efficient system development.

Background Art

[0002]

As a background art in this technical field, there is disclosed in JP-A-2010-140396 (PTL 1).

Along with the spread and evolution of IT technology in recent years, environmental changes tend to be intensified in business utilizing the IT technology. For that reason, vendors and others are required to promptly provide systems in response to such changes in the business environment, such that their customers do not miss business opportunities.

[0003]

Therefore, along with system development, techniques for

executing operation simulations on target systems for development for improving accuracy and efficiency of system development have been proposed. As an example of such conventional techniques, there has been proposed, as a technique related to an system for operation simulation of a computer system, a file matching checking system (refer to PTL 1), in which, in case of inputting a file output by one of a plurality of sequentially executed jobs to another job, a file integrity check system for checking whether integrity of the input/output sequences of the file is appropriate based on preceding/succeeding relationships between the job and the another job, wherein the file matching checking system includes a means of checking integrity for checking the integrity by checking a predetermined status of the file of at least one of the job and the another job and a predetermined status between the job and the another job.

Citation List

Patent Literature

[0004]

PTL 1: JP-A-2010-140396

Summary of Invention

Technical Problem

[0005]

In general system development, deliverables of upstream processes (basic design and detailed design) are gradually refined in downstream processes (manufacturing process and test process), thereby materializing a target system for development.

[0006]

Here, in the upstream processes, processing times for jobs, shell scripts, batch files, programs, etc. in a design are unclear, and it is difficult to estimate the processing times. Therefore, a failure may be found in downstream processes due to erroneous estimation of processing times for jobs and other factors in the upstream processes. In this case, significant rework hours (hours for influence checks, reviews, etc. throughout the upstream processes) becomes necessary. Moreover, upstream processes require significant period and cost, and thus the adverse effect on the overall system development due to the rework hours will increase.

[0007]

Also, as in the prior art, there were cases that, even when prerequisites of jobs in downstream processes (after a manufacturing process) were checked, a design error in upstream processes could not be specified and the downstream processes were performed with the design error.

[0008]

Therefore, the present invention is to provide a

technique capable of coping with a problem from a design stage in upstream processes and providing rapid and efficient system development supports.

Solution to Problem

[0009]

According to an aspect of the present invention, there is provided a system development support device including a memory device configured to store design document information in an upstream design process; and a computing device configured to, when a processing time in the case of executing a job, which is scheduled to be executed on a target system for development indicated by the design document information, on an execution device having a predetermined performance does not satisfy a predetermined design specification, sequentially execute respective processes including rearranging a job execution sequence in the design document information, changing the execution device, and changing specification of the execution device according to the success or failure of the satisfaction.

[0010]

According to another aspect of the present invention, there is provided a system development support method using an information processing device including a memory device that stores design document information in an upstream design

process, wherein when a processing time in the case of executing a job, which is scheduled to be executed on a target system for development indicated by the design document information, on an execution device having a predetermined performance does not satisfy a predetermined design specification, the information processing apparatus sequentially executes respective processes including rearranging a job execution sequence in the design document information, changing the execution device, and changing specification of the execution device according to the success or failure of the satisfaction.

Advantageous Effect of Invention

[0011]

According to the present invention, it is possible to cope with problems from a design stage in upstream processes and provide rapid and efficient system development supports.

Brief Description of Drawings

[0012]

[Fig. 1] Fig. 1 is a diagram showing a functional configuration example of a system development supporting device according to an embodiment.

[Fig. 2] Fig. 2 is a diagram showing a hardware configuration example of a system development supporting device according to an embodiment.

[Fig. 3] Fig. 3 is a diagram showing a first flow example of a system development supporting method according to an embodiment.

[Fig. 4] Fig. 4 is a diagram showing a data configuration example of a job design table according to an embodiment.

[Fig. 5] Fig. 5 is a diagram showing a data configuration example of a hardware specification table according to an embodiment.

[Fig. 6] Fig. 6 is a diagram showing a data configuration example of an estimated result table according to an embodiment.

[Fig. 7] Fig. 7 is a diagram showing a second flow example of a system development supporting method according to an embodiment.

[Fig. 8] Fig. 8 is a diagram showing a data configuration example of an estimation interruption flag table according to an embodiment.

[Fig. 9] Fig. 9 is a diagram showing a third flow example of a system development supporting method according to an embodiment.

[Fig. 10] Fig. 10 is a diagram showing a fourth flow example of a system development supporting method according to an embodiment.

[Fig. 11] Fig. 11 is a diagram showing a fifth flow example of a system development supporting method according to an

embodiment.

[Fig. 12] Fig. 12 is a diagram showing a data configuration example of a processing model table according to an embodiment.

[Fig. 13] Fig. 13 is a diagram showing a sixth flow example of a system development supporting method according to an embodiment.

[Fig. 14] Fig. 14 is a diagram showing a data configuration example of an operation record table according to an embodiment.

Description of Embodiments

[0013]

--- A configuration example of a system development supporting device ---

Embodiments of the present invention will be described below in detail with reference to the drawings. Fig. 1 is a diagram showing a functional configuration example of a system development supporting device 100 according to an embodiment. The system development supporting device 100 shown in Fig. 1 is a computer capable of coping with problems from a design stage in an upstream process and providing rapid and efficient system development support.

[0014]

The system development supporting device 100 according

to an embodiment includes a client function 101, a job analysis function 102, and a job execution function 103 as functional configuration thereof. These functions are implemented by the resources of a general information processing apparatus. Also, it is assumed that the resources constituting these functions are connected in a mutually communicable state. For example, the client function 101 is implemented by an input device, and the job analysis function 102 and the job execution function 103 are implemented by a computing device and a memory device.

[0015]

A user of the system development supporting device 100 can perform an input operation with respect to the job analysis function 102 of the system development supporting device 100 through the client function 101 from among these functions. Specifically, the user can designate columns of a job design table 112, a hardware specification table 113, a processing model table 114, an operation record table 115, an estimated result table 116, and an estimation interruption flag table 117 through the client function 101. In this case, the user may work on settings via the client function 101 in any manner.

[0016]

The job analysis function 102 is a substantial function of the system development supporting device 100. Therefore, the job analysis function 102 executes a system development supporting method according to an embodiment by appropriately

using the job design table 112, the hardware specification table 113, the processing model table 114, the operation record table 115, the estimated result table 116, and the estimation interruption flag table 117.

[0017]

In addition, the job execution function 103 actually executes a job in a target system for development and stores an operation record at that time in the operation record table 115.

[0018]

The system development supporting device 100 having such components may have the configuration as shown in Fig. 2. The system development supporting device 100 of Fig. 2 at least includes a memory device 11 including a suitable non-volatile memory element like an SSD (Solid State Drive) or a hard disk drive, a memory 13 including a volatile storage element like a RAM, a computing device 14 like a CPU that loads a program 12 or the like stored in the memory device 11 to the memory 13 to perform overall control of the system development supporting device 100 and various determinations, computations, and control processing, an input device 15 for accepting key inputs or voice inputs from a user, and an output device 16 like a display for displaying data.

[0019]

In the memory device 11, in addition to the program 12

for implementing the functions required as the system development supporting device 100 according to an embodiment, that is, the client function 101, the job analysis function 102, and the job execution function 103, the job design table 112, the hardware specification table 113, the processing model table 114, the operation record table 115, the estimated result table 116, and the estimation interruption flag table 117 are stored.

[0020]

--- Flow of System Development Support ---

Hereinafter, the actual procedure of a system development supporting method according to an embodiment will be described with reference to the drawings. Various operations corresponding to the system development supporting method described below are realized by a program that the system development supporting device 100 loads into a memory or the like and executes. This program includes codes for performing various operations to be described below.

[0021]

Fig. 3 is a diagram showing a first flow example of a system development supporting method according to an embodiment. Here, it is assumed that a user who is a person in charge of system development inputs a processing command optimized for a job through the client function 101 described above. There are no restrictions on the format, the setting

method, the location, etc. of parameters regarding the command.

[0022]

Upon receiving the command, the job analysis function 102 of the system development supporting device 100 executes a process for the preparation of calculating a processing time (step 0801). In this case, the job analysis function 102 searches a record in the job design table 112 based on indicated by the above command information (for example, an operation name, an execution device name, etc.) and extracts values regarding execution devices 11210 from the record.

[0023]

As illustrated in Fig. 4, the job design table 112 includes a collection of records in which, based on operation IDs 11201 indicating layer relationship in the layer structure of jobs as a key, the records including layers 11202, operation names 11203 indicating name of jobs, scheduled start times 11204 of corresponding jobs, scheduled end times 11205, preceding jobs 11206 indicating jobs are set to have execution sequences earlier than corresponding jobs, prerequisite jobs 11207 indicating jobs to be completed prior to corresponding jobs, numbers of pieces of data 11208 to be processed by corresponding jobs, record sizes 11209 of data to be processed by corresponding jobs, execution devices 11210, processing models 11211 showing types of corresponding jobs, and degrees of influence 11212 at the time of rearranging sequences of

corresponding jobs are associated with one another.

[0024]

The values of the degrees of influence 11212 at the time of rearranging sequence are values used for avoiding inconsistency (for example, a file transmission processing before execution of a file creation processing) during rearrangement of a job sequence.

[0025]

The job analysis function 102 in step 0801 searches for a record in the hardware specification table 113 by using extracted values of the execution devices 11210 as a key. In this case, the job analysis function 102 extracts the values of CPU performance 11303 and I/O performance 11304 from the record specified by the search.

[0026]

As shown in Fig. 5, the hardware specification table 113 includes a collection of records in which, based on the values of execution devices 11301 serving as IDs for uniquely identifying execution devices for jobs as a key, the records including device types 11302 of the execution devices, the CPU performance 11303 as computing performance, the I/O performance 11304 as input/output performance, and priorities 11305 at the time of changing execution devices are associated with one another.

[0027]

In addition, the job analysis function 102 sets values of the execution devices 11210, values of the CPU performance 11303, and values of the I/O performance 11304 obtained above as the values of execution devices 11611, the values of CPU performance 11612, and the values of I/O performance 11613 of the estimated result table 116, respectively.

[0028]

A specific example of the estimated result table 116 is shown in Fig. 6. The estimated result table 116 is a table for storing processing times of respective jobs in the target system for development estimated based on information specified by the user or information of the job design table 112, which is design document information.

[0029]

The data configuration of the estimated result table 116 includes a collection of records in which, based on operation IDs 11601 uniquely identifying jobs as a key, values including layers 11602 of the jobs, operation names 11603, scheduled start times 11604, scheduled end times 11605, processing times 11606, preceding jobs 11607, prerequisite jobs 11608, number of pieces of data 11609, record sizes 11610, execution devices 11611, CPU performance 11612, I/O performance 11613, and processing models 11614 are associated with one other. Basically, each of columns other than the processing times 11606 is the same as those in the configuration of the job design

table 112.

[0030]

The job analysis function 102 executes the processing of step 0801 in relation to all of the records in the job design table 112 and outputs obtained values to corresponding records in the estimated result table 116.

[0031]

Subsequently, the job analysis function 102 executes a processing time estimation process (step 0802). The processing time estimation process will be described based on the flow of Fig. 7. In this flow, the job analysis function 102 repeatedly executes the processing of step 0902 below in relation to all the records of the estimated result table 116 (i.e., jobs) (steps 0901 to 0903).

In this case, the job analysis function 102 reads a record from the estimated result table 116, applies information indicated by the corresponding record to a predetermined calculation formula to calculate the processing time of a corresponding job, and stores a value of the processing time calculated here in the processing times 11606 of the record of the corresponding job in the estimated result table 116. Also, the job analysis function 102 calculates a value of the scheduled end time 11605 by adding the value of the processing time to the scheduled start time 11604 and updates the estimated result table 116 (step 0902).

[0032]

Here, the formula for calculating the processing time is the number of pieces of data of a corresponding job $\times$ a record size $\times ((1/\text{CPU performance}) \times \text{CPU performance dependency coefficient} + (1/\text{I/O performance}) \times \text{I/O performance dependency coefficient})$.

[0033]

The description will return to the flow of Fig. 3. Subsequently, the job analysis function 102 compares the scheduled end time 11605 stored in the estimated result table 116 with the time in step 802 with the scheduled end time 11205 (the job design table 112) related to the corresponding job and checks whether the scheduled end time has expired (step 0803).

[0034]

In this case, the job analysis function 102 shifts the process to a file output processing of a processing time estimation result (step 0809) when the above-mentioned end time 11605 is the same as or before the scheduled end time 11205 of the job design table 112 or whether an interruption flag 11701 of the estimation interruption flag table 117 (Fig. 8) is "0" (step 0803: end time is within a schedule).

[0035]

On the other hand, when neither the above-mentioned the end time 11605 is the same as or before the scheduled end time

11205 of the job design table 112 nor the interruption flag 11701 of the estimation interruption flag table 117 is "0" (step 0803: the scheduled time expired), the job analysis function 102 shifts the process to a job sequence rearranging process (step 0804).

[0036]

It should be noted that the interruption flag 11701 of the estimation interruption flag table 117 illustrated in Fig. 8 is defined in advance before the start of the flow described above. The estimation interruption flag table 117 controls whether to perform an enhancement process (a job sequence rearranging process, an execution device changing processing, and a hardware specification changing processing) when the scheduled end time 11605 of a job estimated in step 0802 has passed the time indicated by the scheduled end time 11205.

[0037]

When a determination is to be made in step 0803, the job analysis function 102 performs following branched determinations based on the value of the interruption flag 11701.

[0038]

When an Interruption flag is "0": no enhancement processing is performed (first end time expiration check: determined in step 0803).

[0039]

When the interruption flag is set to "1": a job sequence rearranging process is performed (second end time expiration check: determined in step 0805).

[0040]

When the interruption flag is set to "2": a job sequence rearranging process and an execution device changing process are performed (third end time expiration check: determined in step 0807).

[0041]

When the interruption flag has "other" values: a job sequence rearranging process, an execution device changing process, and a hardware specification changing process are performed.

[0042]

Subsequently, the job analysis function 102 executes a job sequence rearranging process (step 0804). The job sequence rearranging process is a process in which a job having the largest value of the layers 11602 in the records of the estimated result table 116 (Fig. 6) is defined as the bottommost layer and a job sequence pattern is appropriately rearranged sequentially from the bottommost layer job to a topmost job (that is, a job having the smallest value of the layers 11602), such that the estimated end time that varies in association with the rearrangement is aimed to be at least the same as the scheduled end time 11205 of the job design table 112.

[0043]

The job sequence rearranging process will be described below in detail based on the flow of Fig. 9. In this flow, the job analysis function 102 sets a record corresponding to the largest value of the layers 11602 from among the records of the estimated result table 116 as the bottommost layer of a job and initiates a loop for repeating a job sequence rearrangement for each layer sequentially from a job of the bottommost layer to a job of the topmost layer (that is, one corresponding to the smallest value of the layers 11602) (step 1001).

[0044]

Next, the job analysis function 102 executes a job sequence rearrangement process (step 1002). In this job sequence rearrangement process, the job analysis function 102 changes the value of the preceding job 11607 to a value of the prerequisite job 11608 for a record, which is a record corresponding to the layer 11602 selected during a job layer checking loop (step 1001) (in the initial loop, the record corresponding to the largest value of the layers 11602) and is a record having set thereto the value of the prerequisite job 11608 (not NULL), in the estimated result table 116.

[0045]

In the case of the estimated result table 116 shown in Fig. 6, out of records having operations IDs 11601 "1-2" and

"1-3", the value "1-2" of the preceding job 11607 of the record having the operation ID "1-3" to which a value is set to the prerequisite job 11608 is changed to "1-1", which is the value of the prerequisite job 11608.

[0046]

Subsequently, the job analysis function 102 performs a calculation of the processing time 11606 again with the sequence pattern rearranged in the job sequence rearrangement (step 1002) in the same regard as in step 0902 described above and stores a result of the calculation in the processing time 11606 of the estimated result table 116. At the same time, the job analysis function 102 calculates the scheduled end time 11605 by adding a result of the calculation of the processing time 11606 to the scheduled start time 11604 and updates a corresponding column of the estimated result table 116 (step 1003).

[0047]

In addition, the job analysis function 102 checks whether an end time has passed the scheduled end time 11605 obtained in step 1003 in the same regard as in step 0803 described above (step 1004).

[0048]

As a result, when the scheduled end time 11605 of the estimated result table 116 is the same as or before the scheduled end time 11205 of the job design table 112 (the step

1004: the end time is within the schedule), the job analysis function 102 terminates the flow.

[0049]

On the other hand, as a result of the check, when the scheduled end time 11605 of the estimated result table 116 is not the same as or before the time of the scheduled end time 11205 of the job design table 112 (step 1004: the scheduled time expired), the job analysis function 102 returns to the job layer check loop (step 1001) and performs steps 1002 to 1004 described above with respect to a record of a job corresponding to a layer that is one layer higher than the layer of the processed job, and terminates the flow with termination of the job layer check loop (step 1005).

[0050]

The description will return to the flow of Fig. 3. In response to a result of the above job sequence rearranging process (step 0804), when the scheduled end time 11605 of the estimated result table 116 is the same as or before the scheduled end time 11205 of the job design table 112 or when the value of the interruption flag 11701 of the estimation interruption flag table 117 is "1" (step 0805: end time is within the schedule), the job analysis function 102 proceeds to file output processing of a processing time estimation result (step 0809).

[0051]

On the other hand, when neither of the conditions is satisfied (step 0805: scheduled time expired), the job analysis function 102 proceeds to an execution device changing process (step 0806).

[0052]

Next, the job analysis function 102 executes the execution device change process (step 0806). In the execution device change process, for a job whose scheduled end time 11605 of the estimated result table 116 has passed the scheduled end time 11205 of the job design table 112, the execution device 11611 is replaced with a higher performance execution device, such that the condition that the scheduled end time 11605 of the estimated result table 116 is aimed to become the same as or before the scheduled end time 11205 of the job design table 112.

[0053]

The scheme for processing the execution device change process will be described based on the flow of Fig. 10. In this flow, the job analysis function 102 performs an execution device change target extraction process (step 1101). In step 1101, the job analysis function 102 identifies a record in which the scheduled end time 11605 of the estimated result table 116 is not the same as or before the scheduled end time 11205 of the job design table 112. Here, it is assumed that a record corresponding to, for example, the operation ID "1" is

identified in the estimated result table 116 of Fig. 6.

[0054]

Next, the job analysis function 102 extracts the operation ID 11601 from the record that a value of the layer 11602 is greater than that of the record corresponding the operation ID "1" identified in step 1101 and a value is stored in the execution device 11611 (step 1102). Here, it is assumed that each of the records corresponding to the operation IDs "1-1", "1-2", and "1-3" are identified in the estimated result table 116 of Fig. 6.

[0055]

Subsequently, the job analysis function 102 performs an execution device change process (step 1103). In this step, the job analysis function 102 refers to the hardware specification table 113 by using a value of the execution device 11611 associated with the operation ID 11601 extracted in step 1102 (for example, "C") as a key and determines that the device type 11302 of the execution device 11611 is "AP", for example. Also, the job analysis function 102 identifies an execution device of the same device type as the device type 11302 identified above in the hardware specification table 113. In the case of the hardware specification table 113 shown in Fig. 5, execution devices "A" and "B" whose device type 11302 is "AP" can be identified.

[0056]

In addition, in relation to the execution device "C" corresponding to the operation ID 11601 that is set as an execution device change target in step 1102, the job analysis function 102 identifies the execution device "B" from the execution devices "A" and "B" identified as described above as an execution device having a value "2", which is obtained by subtracting "1" from the value "3" of the priority 305 at the time of changing the execution device, as the value of the priority 305.

[0057]

The job analysis function 102 sets the execution device "B" identified above to the execution device 11611 of each of records corresponding to the operation IDs "1-1", "1-2", and "1-3" and changes the initial execution device "C" to the execution device "B". At this time, the values of the CPU performance 11612 and the I/O performance 11613 are updated to the value of the execution device "B" (extracted from the hardware specification table 113).

[0058]

Next, the job analysis function 102 calculates an end time (step 1104). During calculation of the end time, the job analysis function 102 calculates the value of the processing time 11606 again based on values of the execution device 11611 changed in step 1103, that is, the CPU performance 11612 and the I/O performance 11613 of the execution device "B". In

addition, the job analysis function 102 calculates the scheduled end time 11605 again by adding the value of the processing time 11606 calculated again to the scheduled start time 11604 and updates the estimated result table 116.

[0059]

Subsequently, the job analysis function 102 checks whether an end time has passed a scheduled end time in the same regard as in step 0803 described above (step 1105). In this case, when the scheduled end time 11605 of the estimated result table 116 is the same as or before the scheduled end time 11205 of the job design table 112 (step 1105: the end time is within the schedule), the job analysis function 102 terminates the flow.

[0060]

On the other hand, when the scheduled end time 11605 of the estimated result table 116 is not equal to or before the scheduled end time 11205 of the job design table 112 (step 1105: scheduled time expired), the job analysis function 102 proceeds to a process for checking priority at the time of changing an execution device (step 1106).

[0061]

Subsequently, the job analysis function 102 checks the priority at the time of changing an execution device (step 1106). In this case, with respect to the operation ID 11601 corresponding to expiration of the scheduled end time 11605

that is found when it is checked in step 1105 whether the end time has passed the scheduled end time, the job analysis function 102 refers to the hardware specification table 113 by using a value of the execution device associated with the corresponding operation ID 11601 as a key. As a result of the reference, when it is determined that values of the priorities 305 of the execution devices 11301 of the same device type 11302 at the time of changing an execution device are all "1" (step 1106: all values are 1), the job analysis function 102 terminates the flow.

[0062]

On the other hand, as a result of the above-mentioned reference, when it is determined that the values of the priorities 305 at the time of changing an execution device do not satisfy the condition that all of the values are 1 (step 1106: not all values are 1), the job analysis function 102 returns to the first execution device change target extraction (step 1101).

[0063]

The description will return to the flow of Fig. 3. As described above, the job analysis function 102 that has completed the execution device changing process (step 0806) checks whether an end time has passed a scheduled end time in the same regard as in step 0803 described above (step 0807). In this case, when the scheduled end time 11605 of the estimated

result table 116 is the same as or before the scheduled end time 11205 of the job design table 112 or when the interruption flag 11701 of the estimation interruption flag table 117 is "2" (step 0807: the end time is within the schedule), the job analysis function 102 proceeds to the file output process of the processing time estimation result (step 0809).

[0064]

On the other hand, when neither the scheduled end time 11605 of the estimated result table 116 is the same as or before the scheduled end time 11205 of the job design table 112 nor the interruption flag 11701 of the estimation interruption flag table 117 is "2" (step 0807: the end time is within the schedule), the job analysis function 102 proceeds to the hardware specification changing process (step 0808).

[0065]

Next, the job analysis function 102 performs a hardware specification changing process (step 0808). The hardware specification changing process is a process for, with respect to a job that the scheduled end time 11605 of the estimated result table 116 has passed the scheduled end time 11205 of the job design table 112, identifying one with the longest processing time from among lower jobs constituting the corresponding job, reducing the processing time 11606 of the corresponding job, and changing the hardware specifications (CPU performance or I/O performance) of the execution device

11611 of the corresponding job to a necessary specification, such that the scheduled end time 11605 is within the scheduled end time 11205 of the job design table 112, which is a scheduled end time on the design.

[0066]

The scheme for processing the hardware specification change process will be described based on the flow of Fig. 11. In this flow, the job analysis function 102 extracts a record whose scheduled end time 11605 of the estimated result table 116 is not the same as or before the scheduled end time 11205 of the job design table 112 (step 1201). Here, for example, it is assumed that records corresponding to the operation IDs "1" and "2" have been extracted.

[0067]

Next, the job analysis function 102 calculates times by which the scheduled end time 11605 of the records corresponding to the operation IDs 11601 "1" and "2" extracted in step 1201 deviate from the scheduled end time 11205 of the job design table 112 (deviation time), identifies one with the largest deviation time, that is, the one corresponding to the scheduled end time 11605 most diverged from the scheduled end time 11205 of the job design table 112 as an operation ID "1", for example, and extracts the processing time thereof (step 1202).

[0068]

Next, the job analysis function 102 extracts the

operation ID 11601 corresponding to a value of the layer 11602 greater than "1" of the operation ID 11601 identified in step 1202, that is, the operation ID 11601 of a lower layer, in which a value is stored in the execution device 11611 associated with the corresponding operation ID (operation 1203). Here, for example, it is assumed that operation IDs "1-1", "1-2", and "1-3" could be extracted.

[0069]

Next, the job analysis function 102 identifies, from among the operation IDs extracted in step 1203, those to be processed continuously with each other based on the value of the preceding job 11607, and, regarding a series of records identified here (that is, jobs) sums the values of the processing time 11606 thereof and extracts the operation ID 11601 corresponding to the largest total value of the processing time 11606 (step 1204).

[0070]

For example, when the operation IDs "1-1" and "1-2" constitute a series of records and the operation IDs "1-1" and "1-3" constitute another series of records, the processing time of the operation IDs "1-1" and "1-2" is calculated as $40 + 60 = 100$ and the processing time of the operation IDs "1-1" and "1-3" is calculated as $40 + 90 = 130$. Then, operation IDs "1-1" and "1-3" of the longest processing time "130" out of the total values of the processing time 11606 are extracted.

[0071]

Next, the job analysis function 102 subtracts the deviation time calculated in step 1202 from the longest one of the processing time 11606 of the operation IDs 11601 extracted in step 1204 (step 1205).

[0072]

For example, the longest one of the processing times 11606 of the operation IDs "1 -1" and "1-3" is identified as "90" of the operation ID "1-3", and the deviation time "10" calculated with respect to the operation ID "1" in step 1202 is subtracted from "90" to obtain a processing time "80".

[0073]

Next, the job analysis function 102 uses a calculation formula corresponding to a processing model 11614 associated with the operation ID 11601 identified in step 1205 as a target to be processed to calculate hardware specification (CPU performance 11612 or I/O performance 11613) based on the processing time calculated in step 1205 (step 1206).

[0074]

The calculation formula corresponding to the processing model 11614 is selected according to the following pattern.

[0075]

Pattern 1: In processing model table 114 (Fig. 12), for a processing model that a CPU performance dependency coefficient 11402 is greater than an I/O performance dependency

coefficient 11403, a processing time can be significantly reduced by improving the CPU performance, and thus necessary specification is to be calculated with respect to CPU performance. Of course, when the I/O performance dependency coefficient is greater than the CPU performance dependency coefficient, the processing time may be significantly reduced by improving I/O performance, and thus necessary specification is to be calculated with respect to the I/O performance.

[0076]

In this case, the calculation formula is as follows:
 required CPU performance = number of pieces of data 11609 × record size 11610 × CPU performance dependency coefficient 11402 × I/O performance 11613 / (processing time 11606 × I/O performance 11613 - number of pieces of data 11609 × record size 11610 × I/O performance dependency coefficient 11403).
 The value of the "processing time 11606" in this calculation formula is the value of the processing time calculated in step 1205 (the same applies hereinafter).

[0077]

In this step 1206, the job analysis function 102 stores the required CPU performance calculated as described above in the CPU performance 11612 of the estimated result table 116.

[0078]

Pattern 2: In the processing model table 114, when the CPU performance dependency coefficient 11402 is smaller than

the I/O performance dependency coefficient 11403, as described above, the processing time can be significantly reduced by improving the I/O performance, and thus necessary specification is to be calculated with respect to the I/O performance.

In this case, the calculation formula is as follows:
 required I/O performance = number of pieces of data 11609 × record size 11610 × I/O performance dependency coefficient 11403 × CPU performance 11612 / (processing time 11606 × CPU performance 11612 - number of pieces of data 11609 × record size 11610 × CPU performance dependency coefficient 11402).
 [0079]

In this step 1206, the job analysis function 102 stores the required I/O performance 11614 calculated as described above in the I/O performance 11614 of the estimated result table 116.
 [0080]

Next, based on the value of the CPU performance 11612 or the I/O performance 11613 changed in step 1206 and the estimated result table 116 or the job design table 112, in the same regard as in step 0802, the job analysis function 102 calculates the value of the processing time 11606 again, calculates the scheduled end time 11605 by adding the processing time obtained here to the scheduled start time 11604, and updates the estimated result table 116 (step 1207).

[0081]

Subsequently, the job analysis function 102 checks whether an end time has passed a scheduled end time in the same regard as in step 0803 described above (step 1208). In this case, when the scheduled end time 11605 of the estimated result table 116 is the same as or before the scheduled end time 11205 of the job design table 112 (step 1208: the end time is within the schedule), the job analysis function 102 terminates the flow.

[0082]

On the other hand, when the scheduled end time 11605 of the estimated result table 116 is not equal to or before the scheduled end time 11205 of the job design table 112 (step 1208: scheduled time expired), the job analysis function 102 returns to the step 1201.

[0083]

The description will be return to the flow of Fig. 3. Next, the job analysis function 102 outputs a processing time estimation result, such as the estimated result table 116, to the memory device 11 or a predetermined device or the like as a file (step 0809) and terminates the flow.

[0084]

--- Flow for Updating Dependency Coefficient ---

Next, a process for updating a CPU performance dependency coefficient or an I/O performance dependency coefficient will

be described. Fig. 13 is a diagram showing a sixth flow example of a system development supporting method according to an embodiment.

[0085]

Here, it is assumed that a certain user issues an instruction to update the dependency coefficient through the client function 101. There are no restrictions on the format, the setting method, the location, etc. of parameters regarding the command.

[0086]

The dependency coefficient update process is a process for calculating a CPU performance dependency coefficient and an I/O performance dependency coefficient from values of the processing time 11506 or other columns of the operation record table 115 shown in Fig. 14 and updating the CPU performance dependency coefficient 11402 and the I/O performance dependency coefficient 11403 of the processing model table 114. Each dependency coefficient updated in this process is used to enable more accurate processing time estimation during subsequent processing time estimation processes in the system development supporting device 100.

[0087]

The operation record table 115 is a table that stores actual measured values of start times, end times, and processing times in the case where various types of jobs are

executed in a system with predetermined performance as shown in Fig. 14.

[0088]

In the flow, the job analysis function 102 starts a loop for updating performance dependency coefficients sequentially from topmost layer to the bottommost layer of a processing model 11401 of the processing model table 114 (step 1301).

[0089]

Next, with respect to the processing model 11401 selected in step 1301, the job analysis function 102 identifies records with the same value of the processing model 11211 from among the records of the job design table 112 and extracts the operation ID 11201 corresponding to the records (step 1302).

[0090]

Next, the job analysis function 102 starts a loop for referring to the operation record table 115 by sequentially using the operation IDs 11201 extracted in step 1302 (step 1303).

[0091]

Next, in the operation record table 115, from among the records with the operation IDs 11502 identical to the operation ID 11201 selected in step 1303, the job analysis function 102 extracts records corresponding to an operation ID 11502 of the latest operation date 11501 and the second latest operation date 11501 (step 1304).

[0092]

Next, the job analysis function 102 creates simultaneous equations using the CPU performance dependency coefficient 11402 as a variable "X" and the I/O performance dependency coefficient 11403 as a variable "Y" for the two records extracted in step 1304 and calculates coefficients of the respective records (step 1305). The simultaneous equations in this case are as follows.

[0093]

Equation 1 (data of the latest operation date):

$$\text{processing time} = \text{number of pieces of data 11507} \times \text{record size 11508} \times ((1 / \text{CPU performance 11510}) \times X + (1 / \text{I/O performance 11511}) \times Y)$$

Equation 2 (data of the second latest operation):

$$\text{processing time} = \text{number of pieces of data 11507} \times \text{record size 11508} \times ((1 / \text{CPU performance 11510}) \times X + (1 / \text{I/O performance 11511}) \times Y)$$

Next, the job analysis function 102 calculates an average value of the CPU performance dependency coefficient (variable X) and the I/O performance dependent coefficient (variable Y) calculated for each operation ID in step 1305 (step 1307).

[0094]

Next, the job analysis function 102 updates the values of the CPU performance dependency coefficient 11402 and the I/O performance dependency coefficient 11403 in the processing

model table 114 with the CPU performance dependency coefficient and the I/O performance dependency coefficient calculated in step 1307 (step 1308).

[0095]

The CPU performance dependency coefficient 11402 and the I/O performance dependency coefficient 11403 of the processing model table 114 to be updated in this manner indicate dependency of the processing time on the CPU performance and the I/O performance. By using these coefficients, a processing time can be calculated according to the following equations.

[0096]

$$\text{CPU processing time} = \text{data amount} \times \text{record size} \times \text{CPU performance dependency coefficient} \div \text{CPU performance}$$

$$\text{I/O processing time} = \text{data amount} \times \text{record size} \times \text{I/O performance dependency coefficient} \div \text{I/O performance}$$

By calculating and updating the CPU performance dependency coefficients and the I/O performance dependency coefficients for every predetermined period, in accordance with an event like an increase or decrease in the number of pieces of normal data in a target system, the CPU the performance dependency coefficient and the I/O performance dependency coefficient can be maintained with high precision.

[0097]

Also, in case of an occurrence timing of an event accompanied by an unexpected phenomenon like a sudden increase

or decrease in the number of pieces of data, the operation record may be extracted from the operation record table 115 and the CPU performance dependency coefficient and the I/O performance dependency coefficient may be calculated based on two operation records related to jobs of a predetermined processing model in the extracted operation records.

[0098]

Alternatively, the operation record table 115 may be managed for each event accompanied by the above-mentioned unexpected phenomenon. In this case, during calculation and updating of the CPU performance dependency coefficient and I/O performance dependency coefficient, the job analysis function 102 selects the corresponding operation record table 115 in relation to, for example, an event designated by a user or an upcoming event and, based on two operation records related to a predetermined period and a predetermined processing model, calculates the CPU performance dependency coefficient and the I/O performance dependency coefficient.

[0099]

Although the best modes and the like for carrying out the present invention have been described above in detail, the present invention is not limited thereto and various modifications can be made therein without departing from the gist of the present invention.

[0100]

According to an embodiment, by simulating the operation of a target system for development, the processing time of a job can be estimated step by step from upstream processes, a job sequence can be rearranged or an execution device can be changed according to a result of the estimation, thereby an optimal implementation to job can be supported.

[0101]

As a result, system developers can develop systems with job optimization, improved efficiency, and improved speed from upstream processes, downstream processes to an enhancement after actual system-work. Also, customers of the system developers can quickly use systems that respond appropriately to changes in the business environment.

[0102]

Therefore, it is possible to cope with problems from a design stage in upstream processes and provide rapid and efficient system development supports.

[0103]

By the description of the present specification, at least the following will be clarified. In other words, in a system development supporting device according to an embodiment, at the time of rearranging execution sequence of the jobs, the computing device may identify, among jobs indicated by the design document information, a job whose execution condition is execution of a prerequisite job and which is the same as

another job in terms of the execution condition and defined to be capable of being multiply invoked, and make the execution sequence of the job in the design document information to be identical to that of the another job.

[0104]

Accordingly, a job execution sequence can be rearranged efficiently and accurately, and thus problems can be coped with from a design stage in upstream processes, thereby providing more quick and efficient system development support.

[0105]

In addition, in the system development supporting device according to an embodiment, at the time of changing the execution device, instead of the execution device of a job for which the processing time failed to meet the design specification, the computing device may identify an execution device candidate whose performance is higher than that of the execution device to some extent in a predetermined list of execution devices and change the execution device of the corresponding job defined in the design document information with the execution device candidate.

[0106]

Accordingly, it is possible to efficiently and accurately change an execution device of a job at a minimum cost and cope with problems from the design stage in the upstream processes, thereby providing more quick and efficient

system development support.

[0107]

Further, in the system development supporting device according to an embodiment, at the time of changing the hardware specification of the execution device, the computing device may identify a job with a processing time deviated the least from the design specification, identify specification with which the processing time of the corresponding job becomes a processing time defined by the design specification by applying the content of the identified job to a predetermined formula defining a relationship between the content of a job and an execution device of predetermined specification, and change specification of an execution device of the corresponding job defined in the design document information to the identified specification.

[0108]

Accordingly, it is possible to efficiently and accurately change the specification of an execution device of a job at a minimum cost and cope with problems from the design stage in the upstream processes, thereby providing more quick and efficient system development support.

[0109]

Also, in the system development supporting device according to an embodiment, the computing device may calculate a processing time in the case of executing the job on the

execution device by applying values of the number of pieces of data and record size of the job, the computing performance and the I/O performance of the execution device, and dependencies on the computing performance and the I/O performance during the execution of the job to a predetermined formula and may determine whether the job is successful by comparing the calculated processing time with the design document information.

[0110]

Accordingly, a processing time of a job can be calculated efficiently and accurately and it can be simply determined whether design specification has been met, and thus problems can be coped with from a design stage in upstream processes, thereby providing more quick and efficient system development support.

[0111]

Further, in the system development supporting device according to an embodiment, the computing device may extract values of a processing time, the number of pieces of data, a record size, and the computing performance and the I/O performance of an execution device from each of two operation records related to jobs of the same type from an operation records during a predetermined period for the target system for development and another system having the same configuration as the target system for development, apply the

extracted values of the operation records to the predetermined formula, solve simultaneous equations using dependencies on the computing performance and the I/O performance as variables by using a predetermined algorithm, and calculate the dependencies for every predetermined period.

[0112]

Accordingly, calculation of a processing time of a job and determination of whether design specification has been met can be continuously performed in high precision, and thus problems can be coped with from a design stage in upstream processes, thereby providing more quick and efficient system development support.

[0113]

Furthermore, in the system development supporting device according to an embodiment, at the time of calculating the dependencies, the computing device may extract the values from two operation records that are operation records related to the occurrence period of a predetermined event and are related to jobs of the same type from among operation records during the predetermined period.

[0114]

Accordingly, the dependencies are tuned in a precision in cope with events including a rapid increase or decrease of an amount of data to be processed by a job, and thus calculation of a processing time of a job and determination of whether

design specification has been met can be continuously performed in even higher precision. Therefore, problems can be coped with from a design stage in upstream processes, thereby providing more quick and efficient system development support.

[0115]

Also, in a system development supporting method according to an embodiment, at the time of rearranging execution sequence of the jobs, the information processing apparatus may identify, among jobs indicated by the design document information, a job whose execution condition is execution of a prerequisite job and which is the same as another job in terms of the execution condition and defined to be capable of being multiply invoked, and make the execution sequence of the job in the design document information to be identical to that of the another jobs.

[0116]

In addition, in the system development supporting method according to an embodiment, the information processing device, at the time of changing the execution device, instead of the execution device of a job for which the processing time failed to meet the design specification, may identify an execution device candidate whose performance is higher than that of the execution device to some extent in a predetermined list of execution devices and may change the execution device of the corresponding job defined in the design document information

with the execution device candidate.

[0117]

Furthermore, in the system development supporting method according to an embodiment, at the time of changing the hardware specification of the execution device, the information processing device may identify a job with a processing time deviated the least from the design specification, identify specification with which the processing time of the corresponding job becomes a processing time defined by the design specification by applying the content of the identified job to a predetermined formula defining a relationship between the content of a job and an execution device of predetermined specification, and change specification of an execution device of the corresponding job defined in the design document information to the identified specification.

[0118]

Also, in the system development supporting method according to an embodiment, the information processing device may calculate a processing time in the case of executing the job on the execution device by applying values of the number of pieces of data and record size of the job, the computing performance and the I/O performance of the execution device, and dependencies on the computing performance and the I/O performance during the execution of the job to a predetermined formula and determine whether the job is successful by

comparing the calculated processing time with the design document information.

[0119]

Further, in the system development supporting method according to an embodiment, the information processing device may extract values of a processing time, the number of pieces of data, a record size, and the computing performance and the I/O performance of an execution device from each of two operation records related to jobs of the same type from an operation records during a predetermined period for the target system for development and another system having the same configuration as the target system for development, apply the extracted values of the operation records to the predetermined formula, solve simultaneous equations using dependencies on the computing performance and the I/O performance as variables by using a predetermined algorithm, and calculate the dependencies for every predetermined period.

[0120]

Furthermore, in the system development supporting method according to an embodiment, at the time of calculating the dependencies, the information processing device may extract the values from two operation records that are operation records related to the occurrence period of a predetermined event and are related to jobs of the same type from among operation records during the predetermined period.

Reference Signs List

[0121]

- 11: memory device
- 12: program
- 13: memory
- 14: computing device
- 15: input device
- 16: output device
- 100: system development supporting device
- 101: client function
- 102: job analysis function
- 103: job execution function
- 112: job design table
- 113: hardware specification table
- 114: processing model table
- 115: operation record table
- 116: estimated result table
- 117: estimation interruption flag table

CLAIMS

[Claim 1]

A system development supporting device comprising:

a memory device configured to store design document information in an upstream design process; and

a computing device configured to, when a processing time in the case of executing a job, which is scheduled to be executed on a target system for development indicated by the design document information, on an execution device having a predetermined performance does not satisfy a predetermined design specification, sequentially execute respective processes including rearranging an execution sequence of the job in the design document information, changing the execution device, and changing specification of the execution device according to the success or failure of the satisfaction.

[Claim 2]

The system development supporting device according to claim 1, wherein

at the time of rearranging execution sequence of the jobs, the computing device identifies, among jobs indicated by the design document information, a job whose execution condition is execution of a prerequisite job, and which is the same as another job in terms of the execution condition and defined

to be capable of being multiply invoked, and makes the execution sequence of the job in the design document information to be identical to that of the another job.

[Claim 3]

The system development supporting device according to claim 1, wherein

at the time of changing the execution device, instead of the execution device of a job for which the processing time failed to meet the design specification, the computing device identifies an execution device candidate whose performance is higher than that of the execution device to some predetermined extent in a predetermined list of execution devices and changes the execution device of the corresponding job defined in the design document information with the execution device candidate.

[Claim 4]

The system development supporting device according to claim 1, wherein

at the time of changing the specification of the execution device, the computing device identifies a job with the processing time deviated the least from the design specification, identifies specification with which the processing time of the corresponding job becomes a processing

time defined by the design specification by applying the content of the identified job to a predetermined formula defining a relationship between the content of a job and a processing time in an execution device of predetermined specification, and changes specification of an execution device of the corresponding job defined in the design document information to the identified specification.

[Claim 5]

The system development supporting device according to claim 1, wherein

the computing device calculates, for the job, a processing time in the case of executing the job on the execution device by applying values of the number of pieces of data and record size of the job, computing performance and I/O performance of the execution device, and dependencies on the computing performance and the I/O performance during the execution of the job to a predetermined formula and determines whether the job is successful by comparing the calculated processing time with the design document information.

[Claim 6]

The system development supporting device according to claim 5, wherein

the computing device extracts values of a processing time,

the number of pieces of data, a record size, and computing performance and I/O performance of an execution device from each of two operation records related to jobs of the same type from operation records during a predetermined period for the target system for development and another system having the same configuration as the target system for development, applies the extracted values of the operation records to the predetermined formula, solves simultaneous equations using dependencies on the computing performance and the I/O performance as variables by using a predetermined algorithm, and calculates the dependencies for every predetermined period.

[Claim 7]

The system development supporting device according to claim 6, wherein

at the time of calculating the dependencies, the computing device extracts the values from two operation records that are operation records in an occurrence period of a predetermined event and are related to jobs of the same type from among operation records during the predetermined period.

[Claim 8]

A system development supporting method using an information processing apparatus including a memory device

that stores design document information in an upstream design process, wherein when a processing time in the case of executing a job, which is scheduled to be executed on a target system for development indicated by the design document information, on an execution device having a predetermined performance does not satisfy a predetermined design specification, the information processing apparatus sequentially executes respective processes including rearranging an execution sequence of the job in the design document information, changing the execution device, and changing specification of the execution device according to the success or failure of the satisfaction.

[Claim 9]

The system development supporting method according to claim 8, wherein

at the time of rearranging the execution sequence of the job, the information processing apparatus identifies, among jobs indicated by the design document information, a job whose execution condition is execution of a prerequisite job, and which is the same as another job in terms of the execution condition and defined to be capable of being multiply invoked, and makes the execution sequence of the job in the design document information to be identical to that of the another job.

[Claim 10]

The system development supporting method according to claim 8, wherein

at the time of changing the execution device, instead of the execution device of a job for which the processing time failed to meet the design specification, the information processing apparatus identifies an execution device candidate whose performance is higher than that of the execution device to some predetermined extent in a predetermined list of execution devices and changes the execution device of the corresponding job defined in the design document information to the execution device candidate.

[Claim 11]

The system development supporting method according to claim 8, wherein

at the time of changing the specification of the execution device, the information processing apparatus identifies a job with the processing time deviated the least from the design specification, identifies specification with which the processing time of the corresponding job becomes a processing time defined by the design specification by applying the content of the identified job to a predetermined formula defining a relationship between the content of a job and a

processing time in an execution device of predetermined specification, and changes specification of an execution device of the corresponding job defined in the design document information to the identified specification.

[Claim 12]

The system development supporting method according to claim 8, wherein

the information processing apparatus calculates, for the job, a processing time in the case of executing the job on the execution device by applying values of the number of pieces of data and record size of the job, computing performance and I/O performance of the execution device, and dependencies on the computing performance and the I/O performance during the execution of the job to a predetermined formula and determines whether the job is successful by comparing the calculated processing time with the design document information.

[Claim 13]

The system development supporting method according to claim 12, wherein

the information processing apparatus extracts values of a processing time, the number of pieces of data, a record size, and computing performance and I/O performance of an execution device from each of two operation records related to jobs of

the same type from operation records during a predetermined period for the target system for development and another system having the same configuration as the target system for development, applies the extracted values of the operation records to the predetermined formula, solves simultaneous equations using dependencies on the computing performance and the I/O performance as variables by using a predetermined algorithm, and calculates the dependencies for every predetermined period.

[Claim 14]

The system development supporting method according to claim 13, wherein

at the time of calculating the dependencies, the information processing apparatus extracts the values from two operation records that are operation records in an occurrence period of a predetermined event and are related to jobs of the same type from among operation records during the predetermined period.