(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
21 April 2011 (21.04.2011)

PCT

(10) International Publication Number
WO 2011/045112 A1

(51) International Patent Classification:
G06Q 10/00 (2006.01)

(21) International Application Number:
PCT/EP2010/062325

(22) International Filing Date:
24 August 2010 (24.08.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
09172741.2 12 October 2009 (12.10.2009) EP

(71) Applicant *(for all designated States except US)*: **INTERNATIONAL BUSINESS MACHINES CORPORATION** [US/US]; New Orchard Road, Armonk, New York 10504 (US).

(71) Applicant *(for MG only)*: **COMPAGNIE IBM FRANCE** [FR/FR]; 17 avenue de l'europe, F-92275 Bois-Colombes Cedex (FR).

(72) Inventors; and

(75) Inventors/Applicants *(for US only)*: **SANTOLI, Giulio** [IT/IT]; IBM Italia S.p.A., Via Sciangai, 53, I-00144 Roma (IT). **BARILLARI, Fabio** [IT/IT]; IBM Italia S.p.A., Via Sciangai, 53, I-00144 Roma (IT). **BENEDETTI, Fabio** [IT/IT]; IBM Italia S.p.A., Via Sciangai, 53, I-00144 Roma (IT). **IANNUCCI, Pietro** [IT/IT]; IBM Italia S.p.A., Via Sciangai, 53, I-00144 Roma (IT).

(74) Agent: **BELL, Mark**; Compagnie IBM France, Le Plan du Bois, Département de Propriété Intellectuelle, F-06610 La Gaude (FR).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD,

[Continued on next page]

(54) Title: A METHOD SYSTEM AND PROGRAM TO OPTIMIZE JOB EXECUTION SCHEDULED FROM AND EXECUTED ON EXTERNAL APPLICATION CONTAINERS

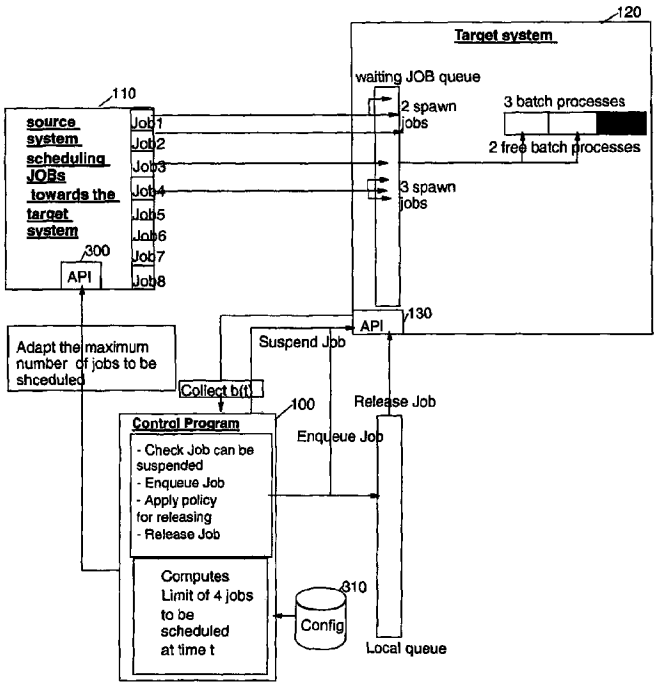

FIGURE 3

(57) Abstract: A method, computer program and system are disclosed which starts from an existing control program which can define an interception rule that can suspend batch jobs scheduled by a source system before their actual execution in a target system, at fixed time interval, the computer program retrieves the list of suspended jobs and, accordingly to user-defined policies, resumes them. The method is enhanced by adding a second level throttling system to optimize use of resources for batch execution in the target system. At a fixed time interval, the control program checks the number of free batch processes in the target system. This value is used to calculate the maximum number ("limit") of jobs that can be concurrently submitted into the target system. Since batch jobs can spawn children jobs during their execution, the combination of the above three actions (to limit concurrent job submission, to define suspension rules, to resume suspended jobs) can be tuned to optimize job execution throughput. All the configuration parameters used in this computer program can be manually defined or can be automatically evaluated by a statistical analysis system, which monitors job execution throughput and adjusts the configuration to get the best performance.

WO 2011/045112 A1

SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) **Designated States** *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *of inventorship (Rule 4.17(iv))*

Published:

— *with international search report (Art. 21(3))*

A METHOD, SYSTEM AND PROGRAM TO OPTIMIZE JOB EXECUTION SCHEDULED FROM AND EXECUTED ON EXTERNAL APPLICATION CONTAINERS

Field of the Invention

The present invention generally relates to batch job execution and more particularly, to management of batch job execution when the jobs are scheduled from and executed on external systems which are application containers.

Background of the Invention

Application containers such as Enterprise Solution Resource Planing (ERP) systems based on SAP, JBoss, ORACLE E-Business suite or IBM Websphere Application Server are able to schedule execution of batch jobs on one other application container (SAP is a trademark or registered trademark of SAP AG in Germany and in several other countries all over the world, JBoss is a trademark of JBoss Inc., Oracle is a registered trademark of Oracle Corporation,IBM, Websphere are trademarks of International Business Machines Corporation in certain countries) .

When submitting batch workload from an originating external system to be executed on a target external system, it is easy to overload the target system and reduce the job execution throughput scheduled from the originating system. The most usual cause is when jobs spawn several children jobs in the target external system, but the maximum number of processes able to run together is fixed in that target system. Moreover, in the target execution system, the management of the job execution queue decreases the overall efficiency and throughput. Some external systems can provide a simple load balancing but cannot manage a big amount of batch submissions.

The external systems considered in the invention are application containers ERP systems such as SAP, JBoss, ORACLE E-Business suite or IBM Websphere Application Server. They contains their own job executor and are able to schedule job execution in one other target external system. They have configuration parameters limiting the number of jobs to be executed simultaneously. All these application containers have an interface which may be GUI, Command Line or an API (such as XBP 2.0 in SAP) to allow control job submission and execution in these systems from the outside.

There is thus a need to control in external systems execution of jobs scheduled from one other external system.

The US patent US 7,231,455 applies to controlling data packets by an external system. It uses a throttling mechanism to limit and control data packets. Even if the external control using a throttling mechanism is efficient, this solution cannot apply to controlling of job execution. The patented solution is for managing data packets transmitted over a network system as the problem raised in the present invention is to manage job execution over more than one scheduler in at least two different systems. The data packets managed by the solution of the prior art have the same size on

both sender and receiver ends : one to one relation as the problem raised in the present invention considers that one job in a scheduler can spread in an unpredictable number of (child) jobs in the other: one to many relations. The solution of prior art is based on cascaded pipeline architecture between the monitored relays. Data goes through this pipeline (of relays). The pipeline acts as a multilevel proxy between the source and destination: both end points do not change their behaviors. It is not the case with the problem raised by the present invention in which the job schedulers to be managed may have different behaviors : as a matter of fact job schedulers in external application containers may be different. Finally, the solution of the prior art is "time period" based: it calculates the amount of data flow in each time period and decides to limit data fan out. This cannot be applied to measure job scheduling capacity which is rather based on the number of background processes available.

The US patent, US7137019 is an IBM patent describing an adaptive throttling system for reducing the impact of non-production work (less and less important) on productive work (more important) in data processing systems by selecting a performance impact limit to satisfy. Even if this patented throttling system looks efficient it cannot be used for solving the problem raised by the present invention: it balances non-production and production work in a data processing system as the present invention raises the problem of managing job execution over more than one scheduler in at least two different systems.

One other US patent US 7,243,121 describes a load balancing scheduling system to control execution of jobs and descendent jobs (children) on a multi-node distributed environment. The method described in US 7,243,121 assumes that is known in advance a given number of children jobs created by a job in execution. In the general case the source scheduling

system is not supposed to know that a job running on the target execution system (for instance SAP) will or will not spawn children jobs and, in any case, the number of children jobs is not determined in advance. US 7,243,121 is mainly a system to split the workload into several pieces and distribute it to different nodes that are under the control of the system. The problem to be solved is rather to optimize the number of jobs between a source scheduling and a target execution system considering that a one job in the source system may match with many jobs on the target system.

A new component provided in IBM Tivoli Workload Scheduler for Application 8.5 allows to control release of jobs into a SAP system (the target external system) according to the current workload of the SAP system. This component is a program interfacing with the external system SAP controlling the scheduling of jobs inside SAP. It is a limiter that ensures that the number of concurrent jobs on SAP does not exceed the maximum resource capacity. This new component uses an API provided by the SAP external system to control job scheduling in the external system.

There is still a need for controlling execution of jobs in target external systems, scheduled from external systems while optimizing resource utilization in the target execution systems.

Summary of the Invention

It is therefore an object of the present invention to have a method for controlling job scheduling performed from one external system in at least one other external system while optimizing resource utilization in the target execution systems.

This object is reached, as claimed in Claim 1 with a method to optimize resources for execution on a target system of batch jobs scheduled from one source system, said batch job generating or not at least one spawned jobs in the target system, said target system being controlled by a control program communicating with the target system through a standard interface, said control program periodically intercepting jobs waiting for execution in the target system and releasing these jobs for execution in the target system according to an evaluation of resource sufficiency in the target system, said method comprising,

- during a periodical control of the target system by the control program, each time a new job has been intercepted by the control program in the target system or, each time all the jobs have been sent on execution in the target system, computing a limit of the number of jobs which can be concurrently scheduled by the source system for execution on the Target system; and,
- providing to the source system, through a standard interface of the source system, the computed limit of the number of jobs which can be scheduled by the source system.

This object is also reached, as claimed in claim 2, with the method of claim 1 wherein the step of providing to the source system a limit of the number of jobs which can be scheduled by the source system comprises:

- reading the amount of resource available at this t time b(t) to execute batch jobs in the target system;
- using α,β,γ and ν parameter values, computing the limit at this t time:

$$Limit(t) = \alpha + \beta \cdot f\left(b(t) + \gamma\right)$$

wherein f is a function providing an average value of the amounts of available resource measured n times.

This object is also reached, as claimed in claim 3, with the method of claim 2 in which the step of computing comprises using as f, the Simple Moving Average of length n, the resulting limit being:

$$Limit^{(v)}_{\alpha,\beta,\gamma}(t) = \alpha + \frac{\beta}{v} \cdot \sum_{k=1}^{v} \left(b(t-k+1) + \gamma \right)$$

This object is also reached, as claimed in claim 4, with the method of claim 2 in which the step of computing comprises using as f, the Exponential Moving Average of length n

$$EMA^{(n)}(x_i) = \eta \cdot x_i + (1-\eta) \cdot EMA^{(n)}(x_{i-1})$$

This object is also reached, as claimed in claim 5, with the method of any one of claim 1 to 4 wherein the α, β, γ and v parameters are read in a configuration file previously updated.

This object is also reached, as claimed in claim 6, with the method of any one of claim 1 to 4 wherein the α, β, γ and v parameter values are computed automatically by the control program using collected statistical data from the use of resources in the target system.

This object is also reached, as claimed in claim 7, with the method of any one of claims 1 to 6 wherein the steps of computing a limit and providing the computed limit are performed on more than one source system.

This object is also reached, as claimed in claim 8, with the method of any one of claims 1 to 7 wherein the steps for communicating with the source system or the target system from

the control program use a standard remote interface provided by the source system or the target system, said control program being executed on a computer remote from the source system or the target system.

This object is also reached, as claimed in claim 9, with the method of any one of claims 1 to 7 wherein the steps for communicating with the source system or the target system from the control program use a standard local interface provided by the source system or the target system, said control program being executed on the same computer as the system providing a local interface.

This object is also reached, as claimed in claim 10, with a computer control program product comprising programming code instructions for executing the steps of the method according to any one of claims 1 to 9 when said program is executed on a computer.

This object is also reached, as claimed in claim 11, with a system comprising means adapted for carrying out the method according to any one of claims 1 to 9.

This solution implements a double level of synchronization, because:

- on one side, the external system is updated for any change in the target execution system and will not submit an overloading number jobs
- on the other side, the target execution system will have an execution queue that does not exceed the maximum number of executable jobs.

The method allows optimizing resource utilization and throughput of job execution in external systems such as application container systems. The method is based on a double level mechanism that allows a fine grained optimization by

manual customization or automatic adaptive tuning. The double level job throttling mechanism of the invention comprises a first level to control children jobs spawned by other running jobs and a second level, new compared to the closest prior art which is the new component provided in IBM Tivoli Workload Scheduler for Application 8.5 which limits the submission of new jobs into an external system.

The method of the invention is "resource" based: it measures the number of background processes available at a given time instant. The principle is one independent entity polling schedulers of the external systems without staying in between mediating data. This entity acts tuning the two schedulers' behaviors by limiting job scheduling on the target execution system and controlling release of jobs sent for execution on the target execution system.

The usual interfaces made available by the external systems are used to control job scheduling in the external systems.

The method of the invention provides the following advantages:

- it does not require changes in the two schedulers only the standard APIs provided by the external systems are used.
- it does not require any change of the connection between the originating external system and the target execution system.
- it is stateless as it does not need historical data or statistics, but it can benefit from statistics data for an automatic tuning.

Brief Description of the Drawings

FIG. 1 illustrates the logical blocks for implementation of a control program according to the method of the prior art;

FIG. 2 illustrates the Control program logical blocks and system environment according to the method of the prior art;

FIG. 3 illustrates the Control program logical blocks and system environment according to the method of the preferred embodiment;

FIG. 4 is the general flowchart of the method of the invention according to the preferred embodiment;

Fig. 5 is a detailed flowchart of the method of the invention according to the preferred embodiment.

Detailed Description of the preferred embodiment

FIG. 1 illustrates the logical blocks of the implementation of the method of the prior art. This method is for controlling job execution in a Target external system (120), for example a SAP system, these jobs having been scheduled from one other external source system (110). The method may be implemented as a computer program (100) that interfaces the Target system through the standard external system programming interfaces (130) such as XBP 2.0 with SAP. The person skilled in the art can adapt this solution to the use by the Control program of different interfaces user interface according to the ability of the Target external system: these interfaces may be remote (programming interfaces, Web User Interface) or local (Command lines, Graphical User Interface); this is why the Control program may be executed on a remote server or on the same server than the external Source or Target systems. As explained in more detail in relation with Fig. 2, the Control program suspends and release Jobs waiting for execution in the Target system. Through the interface, the Control program collects the following information: query list of jobs running on Target

external system, query list of jobs which can be suspended or released in the Target external system, read the number of batch processes in the Target external system, read total number of running jobs on the Target system, read. any required resource on Target system that is needed to run a job, release/resume an enqueued job in the Target system, track the status of running jobs (running, complete) on the Target executing system.

FIG. 2 illustrates the Control program logical blocks and system environment according to the method of the prior art. In the method of the prior art, the Control program (100) takes control of jobs (Job1, Job2 ... Job8) which have been concurrently scheduled by the Source system (110) for their execution in the Target system. The Control program takes control of the Jobs once they have been queued in the waiting queue of the Target system before execution. A single job scheduled can create spawned jobs in the Target system, for instance , for instance Job1 and Job4 have generated respectively 2 jobs and 3 Jobs for execution in the Target system. Using the API of the Target system, the Control program intercepts the Job in the waiting queue, suspends the job and enqueues it in a Local queue. To intercept jobs in SAP, for instance, the SAP XBP2.0 API is used: according to customizable policies, spawned jobs can be suspended until an external program such as the Control program, resumes it. The Control program uses different policies to resume suspended jobs such as:

- resume as many jobs as the total number of batch processes in the SAP system.

- resume as many jobs as the number of free batch processes in the SAP system.

These policies can be associated to criteria needed to prioritize suspended jobs:

- jobs with higher priority must be resumed first
- jobs spawned by the same parent job must be resumed first

It is possible to extend and customize these policies to adapt the control program to the needs of the workload systems.

FIG. 3 illustrates the Control program logical blocks and system environment according to the method of the preferred embodiment. In order to optimize the resources available in the Target system, the algorithm of prior art can adapt the number of jobs to be released in the Target system to the number of free bath processes in the Target system. For instance the Control program can release as many jobs as the number of free batch processes in the Target system. However, in the preferred embodiment is proposed a solution which goes over the best choice of policy to release jobs in the Target system. The idea is to add a control on the Source system. Using the interface capabilities of the Source system, it is possible to add a second level throttling mechanism in order take the best advantage of the resources in the Target system to execute jobs. A new component is added in the Control program for making a computation of the best maximum of jobs to be concurrently scheduled from the Source system to take the best advantage of the resources available in the Target system. In the example of Fig. 3, a maximum of 4 jobs can be scheduled by the Source system. This limit is computed by the Control program and imposed through the available API of the Source system. The control of the maximum of jobs concurrently scheduled by the Source system is done using the available interfaces of the Source system. In the preferred embodiment the Application Programming Interface (300) is used. This limit adaptation (decrease or increase) can thus be done by program automatically at given points in the general algorithm describing the Control program of the method of the preferred embodiment as illustrated in the flowchart of Fig. 4. As with the interface of the Target system, any other interface

available in the Source system and supported by the operating system of the server on which the Control program is running can be used. As detailed in reference to Fig. 5, the computation of the limit of number of jobs to be scheduled depends on the resource available in the target system to executed batch jobs at a given t time and on the value of parameters. The parameter values can be read by the Control program from a configuration file (310) if they are user defined or can be automatically evaluated by a statistical analysis system, which monitors job execution throughput and adjusts the configuration to get the best performance.

It is noted that the computed limit at a given t time is based on the number of resources available for batch job execution at this given time, for instance, the number free batch processes b(t) in SAP Target systems at this given time as explained in more detail in reference to Fig. 5.

FIG. 4 is the general flowchart of the method of the invention according to the preferred embodiment. In the initialization step (400) of Fig. 4, the program controlling both the jobs to be executed in the target executing system and the jobs scheduled from the source scheduling system is started. The controlling program communicates with the external systems using the available interfaces so that the program is dependent on the type of external systems it communicates with. The person skilled in the art can imagine extending the capability of the program to many types of external system and even to more than one interface when they are available on a same target system. For simplification of the description, the detailed description of the preferred embodiment is focusing on one originating Source external system which schedules job into one Target execution external system.

A first part of the flowchart (410, 415, 425, 430, 435, 440), corresponds to the control existing in prior art on jobs

scheduled in the target system performed by an external control program. With this first control, jobs scheduled in the target system are enqueued in a queue local of the control program and released when the program determines that the resources in the target system are available. Three new steps (420, 445, 450) of the preferred embodiment are added, they bring a second control by the control program which applies to the jobs scheduled by the Source scheduling system.

Periodically, with a time period, the program checks (410) if new jobs have been submitted in the Target system. This is done by reading the queue of submitted jobs in the Target system. For instance while using the SAP XBP2.0 API, the interface macros BAPI_XBP_JOB_SELECT and BAPI_XBP_CONFIRM_JOB are used to look into the SAP Job Queue and find new submitted jobs. By getting SAP Job details, it is possible to find out whether the job has been spawn by one other job or not.

New jobs (answer Yes to test 410) are candidate to be controlled by the program if they can be suspended (415). In SAP, this checking is done using the macro BAPI_XBP_GET_INTERCEPTED_JOBS. As a matter of fact, some jobs cannot be suspended: those jobs are submitted from external schedulers or those jobs not are matching the interception criteria stated in SAP Table TBCICPT1. If jobs in the scheduling queue of the target system can be suspended (answer yes to test 415), the program dequeues the jobs from the scheduling queue of the target external system and enqueues them in a queue local to the Control program.

If no job can be suspended among the new jobs scheduled in the target external system (answer No to test 415), the program cannot delay the scheduling for execution of this job but performs a second level of control at the level of the Source scheduling system. The control program changes in the target system the system parameter limiting (420) the number of jobs scheduled in the Target system. This step comprises a computation

of the best limit to put in place according to the number of jobs currently executing and the resources available in the Target system (number of 'free processes' in the example of SAP). The computation is described in Fig. 5.

The program tests if the local queue has jobs to release, (430). This step is also performed if there is no new jobs (answer No to test 410) which have been submitted in the Target system.

If there are jobs to be released from the Local queue of the Control program (answer Yes to test 430), the program checks if there are resources available in the Target system (435). This is done for instance, by using customizable policies such as provided with the SAP XBP2.0 API, spawned jobs can be suspended until an external program (like the Control program described here) resumes them. As described in relation with description of Fig. 2, different policies to resume suspended jobs can be used.

If there is enough resource (answer yes to test 435) the jobs are released (440) from the local queue and re-included in the waiting queue of the Target system. In SAP, this is possible using the marco BAPI_XBP_JOB_START_ASAP.

When there are no more job to control in the Target external system (445) (end of step 440, answer No to test 435 and answer no to test 430) the program changes in the target system the system parameter adapting (450) the number of jobs to be scheduled in the target system. As in step 420, this step comprises a computation of the best limit representing the maximum of jobs to be submitted by the Source system in the Target system. This computation is done knowing the resources available for batch job execution in the Target system. In SAP, the maximum number of concurrent jobs is equal to the number of "SAP Background Processes". In this second execution of computation of limit in the flowchart of the method of the preferred embodiment, as there are resources available, the limit will be increased compared to the result of step 220 which rather

reduces the limit. The computation is described in detail in relation with Fig. 5.

If the program is not stopped (answer Yes to test 455) the process goes on by checking (410) if new jobs have been scheduled by the originating external system in the target system. If not (answer No to test 455) the program ends.

In SAP only jobs that have been internally submitted or any spawned jobs can be intercepted by an external control program. Externally submitted jobs, instead, can not be intercepted. So, when SAP is the target system, the Control program intercepts internally submitted jobs and spawned jobs and enqueue them in the local queue. But this is not the general case and the solution of the preferred embodiment considers the case where any job waiting for execution in the target system can be intercepted and enqueued in the local queue of the control program.

Fig. 5 is a detailed flowchart of the method of the invention according to the preferred embodiment. The steps of this detailed flowchart are executed when the number of job to be scheduled in the Target system is set to a certain limit. These steps are executed when a new job to be executed in the Target system cannot be suspended by the control system (420) and the limit must be decreased to avoid overloading of the Target system. These steps are also executed when a job is released by the control program for execution in the Target system (450) and the limit must be increased to have a better use in the target system of all available resources to execute jobs.

The first step (500) is performed by the control program to check how many free batch processes b(t) are available in the Target system at this t time. Then, the Control program reads (515) in a Configuration file (310) parameters for preparing computation of a formula to provide the maximum number of jobs to be scheduled knowing the value of these

parameters at this t time and the number of free batch processes b(t) which are available in the Target system.

The number of concurrent jobs that can be submitted into the SAP system at time t is given by the $Limit(t)$. This function is calculated as follows:

$$Limit(t) = \alpha + \beta \cdot f\left(b(t) + \gamma\right)$$

where:

- α is a positive or negative constant representing the minimum limit level (if positive) or a fixed negative bias (if negative).
- β is the multiplying factor that translates the number of concurrent SAP jobs into concurrent external jobs.
- γ is a positive or negative constant representing a reserved number of batch processes that must be left always free (if negative) or an additional bias to overload batch (if positive).
- $b(t)$ is the number of SAP free batch process available at time t, and $f(x)$ is a function such as:
  - Simple Moving Average of length n

$$SMA^{(n)}(x_i) = \frac{x_t + x_{t-1} + \mathrm{K} + x_{t-n+1}}{n} = \frac{1}{n}\sum_{k=1}^{n} x_{i-k+1}$$

  - Exponential Moving Average of length n

$$EMA^{(n)}(x_i) = \eta \cdot x_i + (1-\eta) \cdot EMA^{(n)}(x_{i-1})$$

Given configuration parameters α,β,γ and using the Simple Moving Average of length ν, the computed limit value at time t will be:

$$Limit^{(\nu)}_{\alpha,\beta,\gamma}(t) = \alpha + \frac{\beta}{\nu} \cdot \sum_{k=1}^{\nu} \left(b(t-k+1) + \gamma \right)$$

Parameters α,β,γ and ν can be defined in a configuration setting or, optionally, they can evaluated by collecting statistical data on job execution.

If there there are many jobs starting and finishing very quickly, then a slow update of the limit could be better to follow the actual average of the running jobs by ignoring unuseful fluctuations. On the opposite, if there are few slow jobs, then it could be better to change the limit with the same rate as the background processes. Regardless the adopted tuning parameters, the solution of the invention is always able to synchronize the scheduling among Source and Target systems and is also able to avoid that the Target system could get overloaded. If the configuration parameter values are the best ones, then the solution is used at its best capability.

Examples

- For $\alpha = 0, \beta = 1, \gamma = 0, \nu = 1$, the limit will be equal to the number of free batch processes:

$$Limit^{(1)}_{0,1,0}(t) = b(t)$$

- For $\alpha = 0, \beta = 2, \gamma = 0, \nu = 1$, the limit will be twice to the number of free batch processes:

$$Limit^{(1)}_{0,2,0}(t) = 2 \cdot b(t)$$

- For $\alpha = 1, \beta = 1, \gamma = 0, \nu = 1$, the limit will be equal to the number of free batch processes plus one so, even if the number of free batch process is zero, the limit is

$$Limit^{(1)}_{1,1,0}(t) = 1 + b(t)$$

- For $\alpha = 0, \beta = 1, \gamma = -1, \nu = 1$, a batch process is always left free and the limit will be equal to the number of free batch processes less one:

$$Limit^{(1)}_{0,1,-1}(t) = b(t) - 1$$

Even if the solution of the preferred embodiment has been described has been described when one source system schedules batch jobs for execution in one target system, the person skilled in the art can easily adapt the control program to control in the same way one source system scheduling batch jobs for execution in more than one target system: for example, one adaptation could be managing one local queue per each couple (source system, target system).

Claims

1. A method to optimize resources for execution on a target system of batch jobs scheduled from one source system, said batch job generating or not at least one spawned jobs in the target system, said target system being controlled by a control program communicating with the target system through a standard interface, said control program periodically intercepting jobs waiting for execution in the target system and releasing these jobs for execution in the target system according to an evaluation of resource sufficiency in the target system, said method comprising,

- during a periodical control of the target system by the control program, each time a new job has been intercepted by the control program in the target system or, each time all the jobs have been sent on execution in the target system, computing a limit of the number of jobs which can be concurrently scheduled by the source system for execution on the target system; and,
- providing to the source system, through a standard interface of the source system, the computed limit of the number of jobs which can be scheduled by the source system.

2. The method of claim 1 wherein the step of providing to the source system a limit of the number of jobs which can be scheduled by the source system comprises:

- reading the amount of resource available at this t time b(t) to execute batch jobs in the target system;
- using α,β,γ and ν parameter values, computing the limit at this t time:

$$Limit(t) = \alpha + \beta \cdot f\left(b(t) + \gamma\right)$$

wherein f is a function providing an average value of the amounts of available resource measured n times.

3. The method of claim 2 in which the step of computing comprises using as f, the Simple Moving Average of length n, the resulting limit being:

$$Limit^{(v)}_{\alpha,\beta,\gamma}(t) = \alpha + \frac{\beta}{v} \cdot \sum_{k=1}^{v} \left(b(t-k+1) + \gamma \right)$$

4. The method of claim 2 in which the step of computing comprises using as f, the Exponential Moving Average of length n

$$EMA^{(n)}(x_i) = \eta \cdot x_i + (1-\eta) \cdot EMA^{(n)}(x_{i-1})$$

5. The method of any one of claim 1 to 4 wherein the α, β, γ and v parameters are read in a configuration file previously updated.

6. The method of any one of claim 1 to 4 wherein the α, β, γ and v parameter values are computed automatically by the control program using collected statistical data from the use of resources in the target system.

7. The method of any one of claims 1 to 6 wherein the steps of computing a limit and providing the computed limit are performed on more than one source system.

8. The method of any one of claims 1 to 7 wherein the steps for communicating with the source system or the target system from the control program use a standard remote interface provided by the source system or the target system, said control program being executed on a computer remote from the source system or the target system.

9. The method of any one of claims 1 to 7 wherein the steps for communicating with the source system or the target system from the control program use a standard local interface provided by the source system or the target system, said control program being executed on the same computer as the system providing a local interface.

10. A computer control program product comprising programming code instructions for executing the steps of the method according to any one of claims 1 to 9 when said program is executed on a computer.

11. A system comprising means adapted for carrying out the method according to any one of claims 1 to 9.

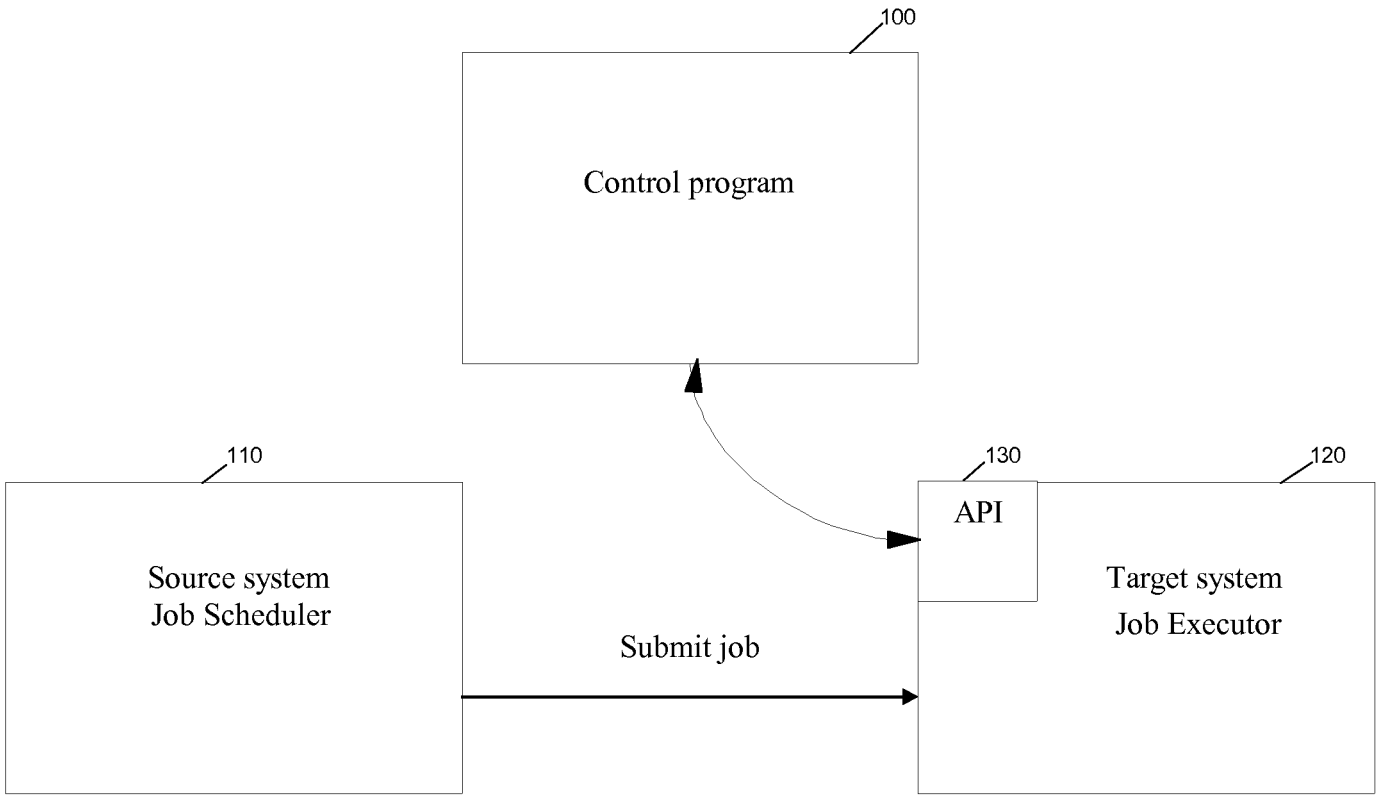
100
Control program
110
Source system
Job Scheduler
130
API
120
Target system
Job Executor
Submit job

FIGURE 1

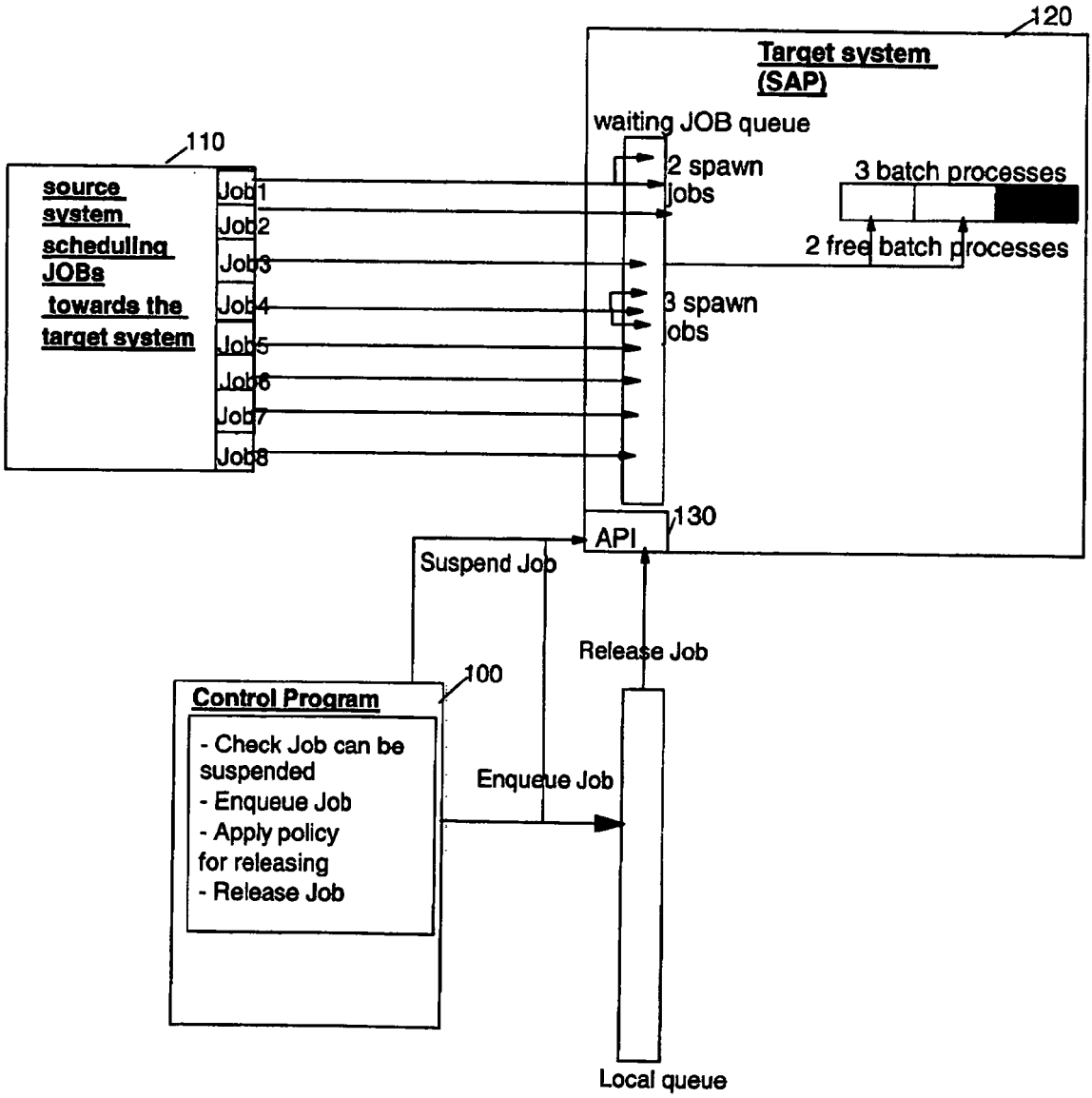
120
Target system
(SAP)
waiting JOB queue
110
source system scheduling JOBs towards the target system
Job1
Job2
Job3
Job4
Job5
Job6
Job7
Job8
2 spawn jobs
3 batch processes
2 free batch processes
3 spawn jobs
130
API
Suspend Job
Release Job
100
Control Program
- Check Job can be suspended
- Enqueue Job
- Apply policy for releasing
- Release Job
Enqueue Job
Local queue

FIGURE 2

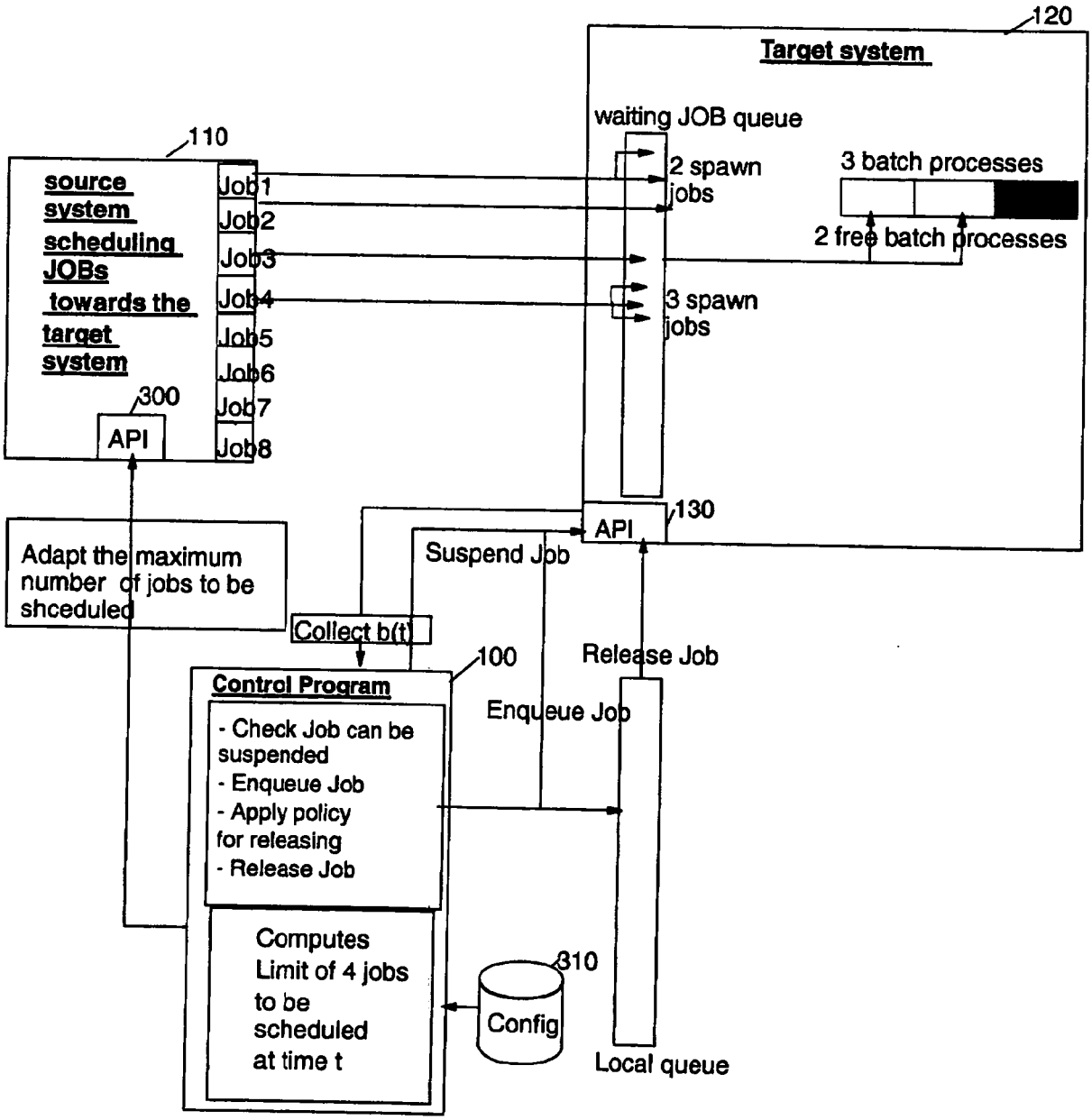
120
Target system
waiting JOB queue
110
source system scheduling JOBs towards the target system
Job1
Job2
Job3
Job4
Job5
Job6
Job7
Job8
2 spawn jobs
3 batch processes
2 free batch processes
3 spawn jobs
300
API
130
API
Adapt the maximum number of jobs to be shceduled
Suspend Job
Collect b(t)
100
Release Job
Control Program
Enqueue Job
- Check Job can be suspended
- Enqueue Job
- Apply policy for releasing
- Release Job
Computes Limit of 4 jobs to be scheduled at time t
310
Config
Local queue

FIGURE 3

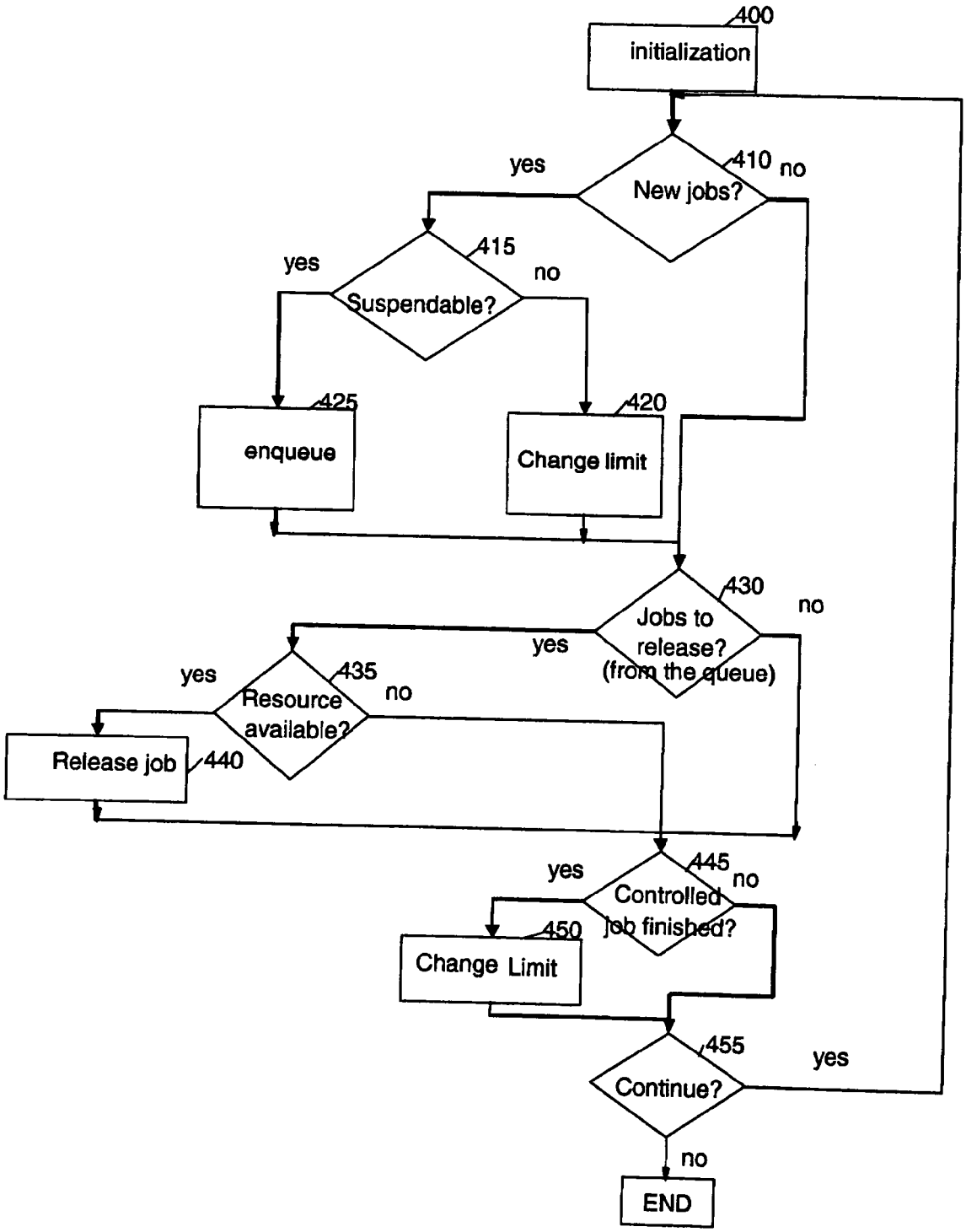

FIGURE 4

SUBSTITUTE SHEET (RULE 26)

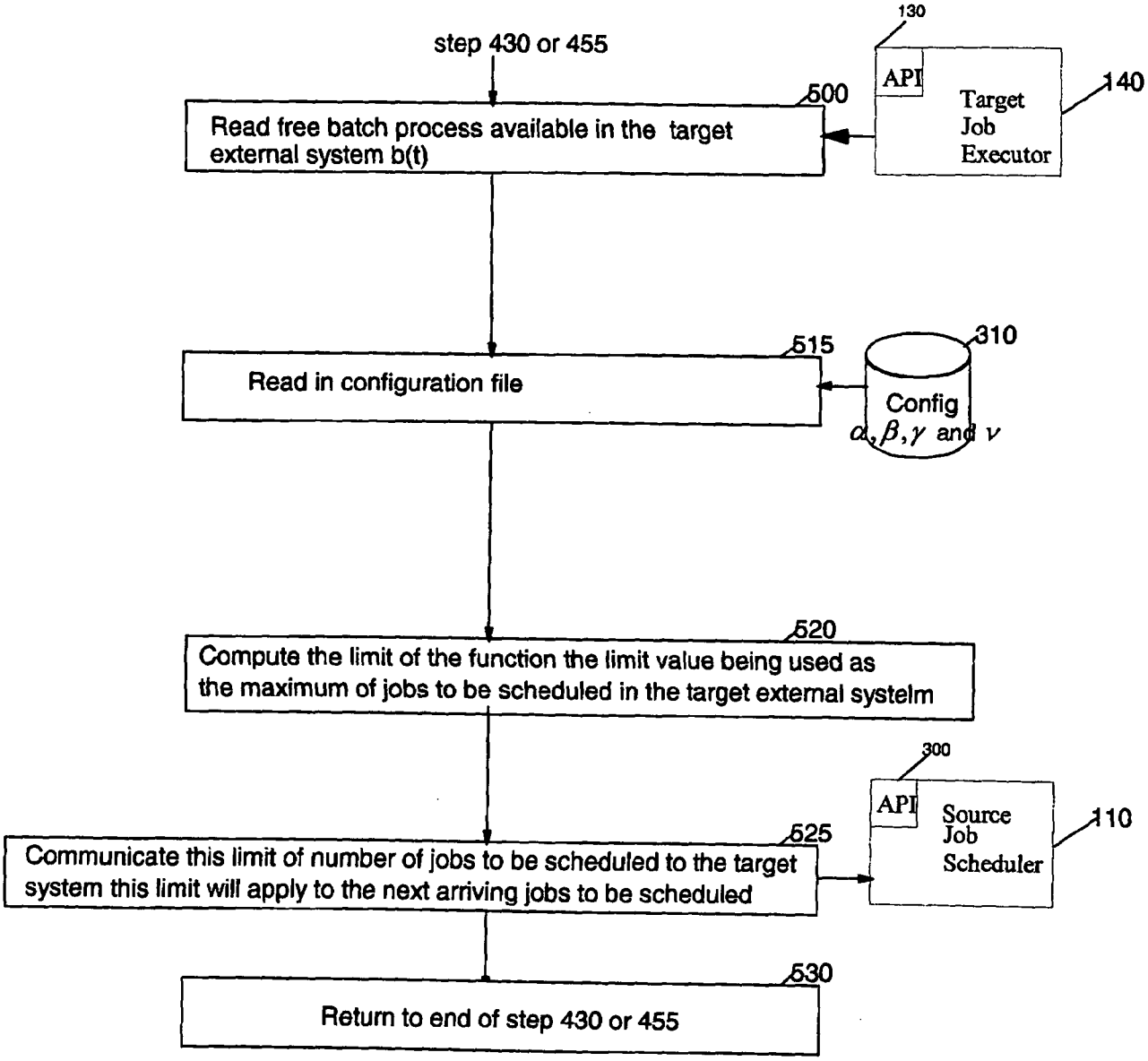
step 430 or 455
130
API
Target Job Executor
140
500
Read free batch process available in the target external system b(t)
515
Read in configuration file
310
Config
α, β, γ and ν
520
Compute the limit of the function the limit value being used as the maximum of jobs to be scheduled in the target external systelm
300
API
Source Job Scheduler
110
525
Communicate this limit of number of jobs to be scheduled to the target system this limit will apply to the next arriving jobs to be scheduled
530
Return to end of step 430 or 455

FIGURE 5

INTERNATIONAL SEARCH REPORT

International application No
PCT/EP2010/062325

A. CLASSIFICATION OF SUBJECT MATTER
INV. G06Q10/00
ADD.

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06Q

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	"STATEMENT IN ACCORDANCE WITH THE NOTICE FROM THE EUROPEAN PATENT OFFICE DATED 1 OCTOBER 2007 CONCERNING BUSINESS METHODS - PCT / ERKLAERUNG GEMAESS DER MITTEILUNG DES EUROPAEISCHEN PATENTAMTS VOM 1.OKTOBER 2007 UEBER GESCHAEFTSMETHODEN - PCT / DECLARATION CONFORMEMENT AU COMMUNIQUE DE L'OFFICE EUROP", 20071101, 1 November 2007 (2007-11-01), XP002456414,	1-11
L	The technical aspects identified in the present application (Art. 15 PCT) are considered part of common general knowledge. Due to their notoriety no documentary evidence is found to be required. For further details see the accompanying Opinion and the reference below.; the whole document	1-11

☐ Further documents are listed in the continuation of Box C.

☐ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
6 December 2010	20/12/2010

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Anastasov, Yuliyan