Internal distribution code:

(A) [-] Publication in OJ
(B) [-] To Chairmen and Members
(C) [-] To Chairmen
(D) [X] No distribution

Datasheet for the decision of 9 November 2021

Case Number: T 0738/17 - 3.5.06

Application Number: 12002449.2

Publication Number: 2518625

IPC: G06F9/54

Language of the proceedings: EN

Title of invention:
High-load business process scalability

Applicant:
SAP SE

Headword:
Process scalability/SAP

Relevant legal provisions:
EPC Art. 56, 84

Keyword:
Inventive step - no

Decisions cited:

Catchword:

Europäisches Patentamt
European Patent Office
Office européen des brevets

Beschwerdekammern

Boards of Appeal

Chambres de recours

Boards of Appeal of the
European Patent Office
Richard-Reitzner-Allee 8
85540 Haar
GERMANY
Tel. +49 (0)89 2399-0
Fax +49 (0)89 2399-4465

Case Number: T 0738/17 - 3.5.06

D E C I S I O N
of Technical Board of Appeal 3.5.06
of 9 November 2021

Appellant: (Applicant)	SAP SE Dietmar-Hopp-Allee 16 69190 Walldorf (DE)
Representative:	Müller-Boré & Partner Patentanwälte PartG mbB Friedenheimer Brücke 21 80639 München (DE)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 11 November 2016 refusing European patent application No. 12002449.2 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairman	M. Müller
Members:	T. Alecu
	B. Müller

Summary of Facts and Submissions

I. The appeal is against the decision of the Examining Division to refuse the application. The appellant requested that the decision under appeal be set aside and a European patent be granted on the basis of the main request or of the auxiliary request, both filed with the statement of grounds of appeal.

II. The main request is identical to the one underlying the appealed decision, wherein it was considered that claim 1 was not compliant with Articles 83 and 84 EPC. The decision also cited in particular the following documents:

D1: EP 1 939 743 A2 (SAP AG [DE]) 2 July 2008
D2: US 2003/041178 A 1 (BROUK LEV [US] ET AL) 27 February 2003
D4: EP 2 196 906 A1 (SAP AG [DE]) 16 June 2010

and found a lack of inventive step of the subject-matter of claim 1 starting from either of D1 or D4.

III. With the communication accompanying a summons to oral proceedings, the Board introduced document

DA1: Doraiswamy S. et al. (2005) Reweaving the Tapestry: Integrating Database and Messaging Systems in the Wake of New Middleware Technologies. In: Härder T., Lehner W. (eds) Data Management in a Connected World. Lecture Notes in Computer Science, vol 3551. Springer, Berlin, Heidelberg. https://doi.org/10.1007/11499923_6.

IV. Claim 1 of the main request defines (reference signs removed, feature lettering by the Board):

A computer implemented method performed by one or more processors for dispatching an event as a message to a business process instance, the method comprising the following operations:

a. receiving a message at a first computer node executing a first business process instance;

b. identifying a second business process instance associated with the message, the second business process instance being assigned to process the message;

c. sending the message, if the second business process instance is not located at the first computer node but at a second computer node other than the first computer node, to a centralized messaging queue for retrieval by the second business process instance, the centralized messaging queue being stored in a memory of a messaging system that is communicably coupled with the first computer node and the second computer node;

d. sending a notification message to the second computer node hosting the second business process instance, the notification message indicating that the message can be retrieved from the centralized messaging queue; and

e. immediately delivering the message, if the second business process instance is located at the first computer node, to the second business process instance, bypassing the centralized messaging queue.

V. Claim 1 of the auxiliary request adds the features of:

f. initiating a polling request from the second computer node to the centralized messaging queue;

identifying a message in the centralized messaging queue for retrieval based on the polling request; retrieving and dequeuing the message from the centralized messaging queue; and
processing the message at the second computer node using a business process instance associated with the message,

g. wherein a plurality of the polling requests to the centralized messaging queue are initiated at a particular time interval to determine whether an incoming message assigned for processing by the second computer node has been received, and

h. wherein an immediate polling request is sent to the centralized messaging queue if a notification is received indicating availability of the message in the centralized messaging queue, wherein the immediate polling request is sent before a subsequent periodic request is to be sent at the particular interval,

i. the method further comprising obtaining a lock on a shared state variable associated with the business process instance before retrieving the message from the centralized messaging queue,

j. wherein obtaining the lock on the shared state variable comprises preventing other components or business process instances other than the business process instance associated with the message from accessing the shared state variable.

VI. Oral proceedings were held as scheduled. The board raised a clarity objection against features i and j of the auxiliary request, which the appellant did not dispute. They indicated that they would be willing to delete the features i and j and asked the Board to provide an opinion on the inventive step of the request without those features. The Board provided that opinion

which was unfavourable. As a consequence, the appellant did not file an additional request with features i and j deleted. At the end of the oral proceedings, the chairman announced the decision of the board.

Reasons for the Decision

The application

1. The application relates to methods and systems for providing high-load business process scalability in cloud/cluster node event-driven infrastructures. The problem to be solved lies in that an event (also called a message) may be generated at a cloud instance/cluster node which is different from the one where the process instance that needs to consume it finds itself (paragraph 13). Instead of directly initiating communication between the sender of an event and a process node, the application proposes to use a centralized queue in which the event is stored until the receiving process instance is able to consume it (paragraph 26). In this way the delivery of the event is decoupled from its reception and consumption.

1.1 The processes poll the queue either at regular intervals or upon notification that an event intended for them has arrived in order to consume the events (paragraphs 61, 68).

1.2 If the process instance is located on the same node as the event, then *the event is immediately delivered to the process instance, bypassing the messaging queue* (paragraph 64).

The prior art

2. Document D1 describes a piece of messaging middleware for business processes. It proposes to replace the standard "centralized" approach of sending events to process specific queues (paragraph 2) with a "decentralized" one, using a global queue which receives all events, codifies the events with properties including, e.g., how many processes need to treat them and whether new process instances need to be generated, and sends them into an internal queue from which the processes consume the events that concern them (paragraphs 3-8, 46-55).

3. Document D2 (abstract, figures 2-3, paragraphs 104-116) describes an open platform for enterprise application integration. It provides a piece of messaging middleware for applications using send and receive queues (with polling and push options) while providing adaptors for data formatting.

4. Document D4 describes what the present application refers to as the "cluster enablement approach" (at paragraphs 22-25). D4 addresses the problem that in a cluster environment an event ("*such as a message, a call, a data object, or the like*" - see paragraph 15) may appear at a node but the process instance for handling it is to be found at a different node (paragraph 11). D4 proposes as a solution to transfer the process instance to the event node. This is realised by "persisting" the process instances into a central database (upon the process becoming idle), so that they can be transferred to the nodes requiring them (paragraphs 15-25).

5. Document DA1 discusses integration of database and messaging systems. It describes message queuing systems as having evolved from basic queuing to being *the backbone* of middleware, using e.g. a database as a data source (page 92, first full paragraph). Features of advanced messaging systems are presented in section 2.1. Sections 3 and 4 present different ways of integrating queue messaging with or within databases.

Main request

Inventive step

6. The Board chooses document D4 as a starting point for the inventive-step analysis. As said above, D4 addresses the issue of an event appearing at a different node than the one where the process instance is; if the event and the process instance find themselves at the same node, then the event is delivered to the process in the "normal" manner (see paragraphs 15 and 30). D4 therefore discloses features a, b and e.

7. However, unlike the current claimed invention (features c and d), D4 transfers the process instance from its current node to the event node. This is done by waiting for the process to be available, transferring and persisting it into a central database, and notifying the event node that the process is ready in the central database, waiting for it to be acquired by the event node (paragraphs 20-25; 31 to 34).

8. When implementing D4, the skilled person would encounter obvious difficulties in moving the process instances, as also explained in the current application at page 24, because they may be too complex, or may be

bound to local resources. So the skilled person would look for alternatives to moving the process instance.

8.1 Considering the issue addressed in D4, they would immediately see that, when a specific process and an event need to be brought together, there are three options: one can move the process to the event node, move the event to the process node, or both to a different node (though this latter solution may be unnecessarily complicated and would not work if the process were tied to a local resource). Thus the skilled person would contemplate moving the events instead of the processes.

8.2 Furthermore, the skilled person knows that events can be, or are treated equivalently to, messages (D4 paragraph 15; D1, paragraph 2), which by definition, may have to be transported between nodes.

8.3 So there are compelling reasons for the skilled person to implement the alternative approach of moving the events instead of the process instances.

9. The Board is of the opinion that, when implementing this alternative, the skilled person would take guidance from D4 and would move events, as much as possible, in the same way as D4 moves processes.

9.1 This means that events will be stored in a centralized database and the process (instead of the event node) will be notified that an event has arrived for handling and is stored in the database.

9.2 Moreover, since events are treated as messages, it is a trivial option to organize them in queues, which is the default way of organizing messages for messaging systems (see D1, D2, DA1), allowing to transmit more than one message concurrently through the messaging system. This is valid both at the level of the central database (as DA1 explains, queues and databases are perfectly compatible; see also the current application at para-

graphs 27, 57, 63), and at the level of the local nodes for sending/receiving events.

9.3 This results in the implementation of the remaining claimed features c and d.

10. The appellant argued that D4 taught a clear request-based method which is specific to moving processes, not events. Changing D4 to transport events instead of processes meant doing away with the teachings of D4. Furthermore, D4 did not hint towards moving events, so the skilled person had no reason, based on D4, to modify D4 to move events instead of processes. In fact, D4 taught away from moving events.

10.1 Also, in D4 (paragraphs 20-25) it was the controller that verified where the corresponding process was, when it was idle, decided to persist it into the database, together with its state variables, and notified the event node. Even if one took the proposed alternative ("reverse mode"), the skilled person would only have considered replacing the controller by a piece of messaging middleware, treating all events, irrespective of whether they found themselves on the same node or not. So feature e was not disclosed or rendered obvious in the reverse case.

10.2 The system of D4, being also request-based, did not require a queue.

11. The Board does not find the arguments of the appellant convincing.

11.1 While it is true that D4 provides no hint towards moving events instead of processes, the skilled person would consider this alternative for the reasons provided above.

11.2 D4 also does not teach *away* from this solution: though D4 teaches a system for transporting processes, it does not, at any point, make any (negative) comment about

the alternative, such as possible difficulties or disadvantages. The Board also does not see in D4 any implied teaching to that effect.

11.3 Regarding feature e, the embodiment of Figure 3 in D4 (paragraph 30 and following) makes clear that the event node determines on which node the process instance is, and requires transport only if necessary. The transport controller (if any, the embodiment of paragraph 3 makes no reference to it), or the corresponding messaging middleware in the alternative, are therefore used only if the event and the process instance are not on the same node: as explained above, the skilled person would implement the alternative in an analogous manner.

11.4 Regarding the fact that D4 itself does not require a queue, the Board considers that the skilled person would also perform the basic adaptations made obvious by the change of paradigm - here one of them is to use a queue when implementing a *messaging* system, as opposed to *a process instance* transport system (for the reasons explained above).

12. The Board concludes that the subject matter of claim 1 of the main request is obvious to the skilled person starting from D4 in view of their common knowledge and thus lacks an inventive step (Article 56 EPC).

Auxiliary request

13. The auxiliary request adds features related on the one hand to messaging features, i.e. a combination of notification and polling for message reception (f to h), and on the other hand to locking shared variables by the process instance (i and j).

14. The second set of features (i and j) is not clear (Article 84 EPC). In particular it is not clear what

the *shared state variable* is, or at least what its function is, and whether there is any functional link between the *shared state variable* and the *message*.

Auxiliary request with features i and j deleted

15. The appellant did not dispute the clarity objection, which was first raised during the oral proceedings before the Board, but indicated that they would be willing to delete the features i and j related thereto and asked the Board to provide an opinion on the inventive step of the request without those features. In the oral proceedings, the Board provided that opinion which was unfavourable. As a consequence, the appellant did not file an additional request with features i and j deleted. (See point VI above.) The unfavourable opinion is explained *obiter* below.

16. The addition (in respect of the main request) of the set of features f to h defines a combination of polling the centralized queue at regular times, by the processes, to ask whether an event is present for consumption by the polling process, with an immediate polling upon notification to the process that an event for the process has arrived at the central queue.

17. Document D4, as discussed above, uses a request-based system. When the process is persisted to the central database, the event node is notified (see figure 3, notifications 324 and 326) and then the process is loaded to that event node. This corresponds to the claimed immediate polling option (feature h) and would be analogously implemented by the skilled person when transferring events instead of processes.

18. The appellant argued that the additional regular polling option (feature g) solves the problem caused by lost notifications (see paragraph 68). There was no hint towards this problem in D4, and the request-based structure of D4 made it that no such regular polling was needed.

19. The Board remarks that the problem of lost notifications is present in the system of D4, whether mentioned or not. For instance, either one of notifications 324 or 326 can be lost, which means that the event node may wait forever.

20. The skilled person would recognize this problem and solve it according to standard procedures - and regular polling is a standard way of verifying whether something has arrived. This is already true for D4 as it is, but it is even more relevant in the discussed alternative: if in the former scheme the regular polling could take place only while waiting to be notified of process availability after a request was made, in the latter the process has no a priori knowledge as to when messages are expected - so continuous regular polling is needed to solve the problem of possible lost message notifications.

21. In the Board's opinion, therefore, expressed *obiter*, the subject matter of the auxiliary request with features i and j deleted lacks inventive step starting from D4 in view of the common general knowledge considering the basic skill set of the skilled person.

Other issues

22. Given the above conclusions, the Board sees no need to decide on the other objections raised by the Examining Division.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

L. Stridde M. Müller

Decision electronically authenticated