Internal distribution code:

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

Datasheet for the decision of 1 July 2020

Case Number: T 2295/14 - 3.5.06

Application Number: 11008415.9

Publication Number: 2444900

IPC: G06F9/54

Language of the proceedings: EN

Title of invention:
Transparent distribution and decoupling of modules using asynchronous communication and scopes

Applicant:
Business Objects Software Ltd.

Headword:
Remote method invocation/BUSINESS OBJECTS SOFTWARE

Relevant legal provisions:
EPC Art. 123(2)

Keyword:
Amendments - added subject-matter (yes)

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 2295/14 - 3.5.06

D E C I S I O N
of Technical Board of Appeal 3.5.06
of 1 July 2020

Appellant: (Applicant)	Business Objects Software Ltd. 1012-1014 Kingswood Avenue Citywest Business Campus Dublin 24 (IE)
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 3 June 2014 refusing European patent application No. 11008415.9 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairman	M. Müller
Members:	S. Krischer
	A. Jimenez

Summary of Facts and Submissions

I. The appeal lies against the decision, dispatched with reasons dated 3 June 2014, refusing European patent application No. 11 008 415.9 for violation of Articles 123(2) and 84 EPC and, as stated in an *obiter dictum*, of Article 56 EPC.

II. A notice of appeal was filed on 4 August 2014, the appeal fee being paid on the same day. A statement of grounds of appeal was received on 13 October 2014, in which the appellant requested that the decision be set aside and a patent be granted on the basis of claims 1-14 according to a main or a first auxiliary request, or claims 1-13 according to a second auxiliary request, all filed with the grounds of appeal. The other application documents are, for all requests, description pages 2, 2a and 3-5 as filed with the grounds of appeal, and description pages 1 and 6-35 and figures 1-10 as originally filed.

III. In an annex to a summons to oral proceedings, the board expressed its opinion that the conclusions of the examining division regarding Articles 123(2), 84 and 56 would have to be confirmed.

IV. In response to the summons, the appellant filed an amended set of claims 1-6 according to a new, sole request.

V. Claim 1 reads as follows:

"A method for decoupling modules (110-4; 120-4; 140-4) and transparently distributing resources in a computer system (100), comprising:

invoking (S201), by a first module (110-4;120-4) of a first host (110;120), a method of an object instance of a second module (140-4) of a second host (140) by performing a local method invocation on an object instance in a local address space of the first module (110-4; 120-4);

allocating a context (110-6) for a scope instance of a request message (110-10; 120-10) on the first host (110; 120), wherein the scope instance defines a period of time, wherein the context (110-6) is a defined storage area including resources (110-7; 110-8) associated with the request message (110-10; 120-10), wherein the resources are in-memory data structures;

queuing (S202), by the first host (110; 120) the request message (110-10; 120-10) and detaching a corresponding thread from a task of executing the method invocation in the first module (110-4; 120-4), wherein the corresponding thread is employed for other tasks on the first host (110; 120);

pausing (S 203) the first module (110-4) for the duration of the method invocation;

transferring (S204) the request message (110-10; 120-10), from the first module (110-4) to the second module (140-4);

sending (S 205), to the second host (140), a copy of the context (110-6; 120-6) of the scope instance associated with the request message;

delivering (S206) the request message to the object instance of the second module (140-4);

executing (S207), by the second host (140), the method of the object instance of the second module (140-4) with reference to the copy of the context;

converting (S208), by the second host (140), a result of the execution of the method into a response message and queuing the response message in a local address space of the second module (140-4);

transferring (S209) the response message to a local address space of the first module (110-4; 120-4);

receiving (S 210), at the local address space of the first module (110-4; 120-4), any changes made to the copy of the context (110-6; 120-6) during the method execution in the second host (140);

replicating the changes into the context (110-6; 120-6);

resuming the first module (110-4; 120-4) on the first host (110; 120); and

releasing the scope instance and the resources of the scope instance at the end of the period of time, wherein the resources of the scope instance include the resources of the context of the scope instance and resources of the copy of the context."

VI. Oral proceedings were held on 1 July 2020 as scheduled. At the end of the oral proceedings, the chairman announced the board's decision.

Reasons for the Decision

The invention

1. The application relates to a "structured approach" for "transparent" communication between (software) modules distributed across different regions or countries, preferably in the context of cloud computing for businesses (see paragraphs 2 and 4; all references are to the application as originally filed). More specifically, the application relates to "decoupling" such modules using what are called asynchronous messaging and "scopes" (see paragraphs 5 and 40).

1.1 The application considers the communication between computer systems (also referred to as "nodes") acting as message "producers" and "consumers" (see figure 1a and paragraphs 41 and 62). Each of them provides a number of "resources", i.e. "anything necessary needed to execute a particular process, for example, in[-]memory data structures" (see paragraph 45).

1.2 "Scopes" - or rather "scope instances" obtained from one of several "scope types" - comprise "a plurality of resources grouped on a computer for a defined period of time" (see again paragraph 45 and table 1 on pages 20-23). Each computer provides a "resource management module" which "associates" resources with scope instances by "allocating them in and releasing them from a" - indeed "exactly one" - "scope instance" (paragraph 46, middle of page 11). This is said to happen "to clearly delineate the lifetime of any resources that are allocated" in a system. It is further disclosed that resources may be allocated to a scope instance when they (presumably the resources) are created and removed when the resources are deleted (still paragraph 46). The resources may also be "released" when the owning scope instance is. The resource management module further associates a scope instance with a "storage area", called its "context" (paragraph 47, first sentence). It is stated that "a context may be distinct from its associated scope instance" (third sentence) - which, in the board's opinion, implies that they might also be identical - and that a "context [...] contains allocated resources" (end of paragraph 47,). It is further stated that "scope instances may migrate" between computers and have a context (or "context

instance") in each (see paragraph 48), but that they may also outlive their "nodes" (see paragraph 62).

1.3 Message passing is said to be asynchronous (see page 13 *et seq.*), although it is also disclosed that the calling module is "paused for the duration of the method invocation". A method invocation at a "message producer" is transformed into a message which is moved from a local "outbound queue" to an "inbound queue" at the message consumer. The "copy of a context for message[s]" may also be transmitted. The sending thread may be suspended or employed with other tasks (paragraphs 49-52, 55 and 72).

1.4 Scope instances are organized in "hierarchies" (see paragraph 60 *et seq.* and figure 5) which may dynamically change at run-time (see paragraph 63). The parent-child relationship must "satisfy the constraint" that, in particular, a "child scope instance does not outlive its parent scope instance": when a parent scope instance is "released", "along with any resources that have been allocated in its context", all its child scope instances are released as well (paragraphs 60, 64 and 65).

The amendments

2. The amended claims were filed in order to overcome, *inter alia*, clarity objections by the board, some of which related to the central concept of the invention, the "scope instance".

3. With regard to this concept, claim 1 specifies in the "allocating" and "releasing" steps that "the scope instance defines a period of time" at the end of which associated resources are released and that the

resources released "at the end of the period of time" includes those of the "copy of the context" used at the second host during the remote method invocation (see claims page 1, lines 8-9 and 24-25, and claims page 2, lines 10-14).

4. The appellant argued in the oral proceedings that the first of these two amendments are based on paragraphs 46 and 65, in combination with table 1.

4.1 The board disagrees. The skilled person would construe the claimed term "scope instance" as an "object" instantiated from a "scope type" and the feature that the scope instance "defines" a period of time as implying that the scope instance would contain sufficient information in terms of its instance variables that a period of time could be derived therefrom; typically, the skilled person would assume the scope instance to have an instance variable storing the relevant time value.

4.2 However, this is not what paragraphs 46, 65 or table 1 disclose.

4.2.1 Paragraph 46 discloses in its first sentence that the "[r]esource management modules [...] may associate resources with scope instances to clearly delineate the lifetime of any resources that are allocated inside [the] system". This expresses "delineation of the lifetime" as the purpose of associating resources with scope instances, but not that it would be the scope instance itself to "define" the lifetime.

4.2.2 Paragraph 65 discloses that the "[system [...] may enforce a defined lifespan for scope instance" but

fails to imply that it is the scope instance to define the lifetime in question.

4.2.3 Table 1, in addition, does not disclose a defined period of time but rather an observed one. It mentions, for instance, in the context of "Method Scope" (i.e. instances of the type "Method Scope") that the duration of the method execution has a "typical duration" of "1 - 50 msec". This does not "define" a lifetime with a fixed end (at which resources can be released), for example because calling a duration "typical" would be understood as implying that there may be exceptions. I.e. there may be methods the execution of which takes longer than 50msec. It is also noted that for some time periods in table 1 only a lower limit is given (see, e.g. "Application Scope").

5. As regards the second of these amendments, the appellant argued in oral proceedings that the original disclosure was to be found in paragraphs 52, 65, 73 and 74 and figures 1B and 7.

5.1 Again, the board disagrees.

5.2 Figures 1B and 7 mention scopes but only indirectly by referring to contexts "of" certain scopes. For instance, they mention a "context of scope 1" (no. 110-6) and a "context of scope 1 (copy)" (nos. 140-7, 710-6) but they do not disclose that the same *scope* or *scope instance* is associated with these two contexts. They only disclose that one context, which is referred to in the figures as "context of scope 1" is copied, and refers to the copy by a different name "context of scope 1 (copy)". The figures not disclosing scopes at all do not exclude that "context of scope 1 (copy)" is associated with a different scope instance,

for instance one local to the message consumer. The disclosure in paragraphs 52 with respect to figure 1B and paragraph 74 with respect to figure 7 are consistent with this interpretation. They disclose the copying of contexts but not which scope instances the copied contexts are associated with for the purpose of resource management.

5.3 Paragraph 73 mentions that contexts have "logical identities" and that a context and its copy has "the same logical identity". The appellant argued with reference to paragraph 45, which discloses that "a scope instance may include a logical ID for identifying this instance during runtime", that the contexts' ID must be the one of the associated scope instance and that, therefore, paragraph 73 disclosed that a context and a copy were associated with the same scope instance.

5.4 The board does not accept this argument. The logical identity of a context must *a priori* be distinguished from that of the scope instance because both may be distinct from each other (see paragraph 47, sentence 3). While it is not impossible that the IDs are used as the appellant suggests, this use is not directly and unambiguously derivable from the application as originally filed.

5.5 Finally, paragraph 65 discloses that "at the end of [a scope instance's] life", "resources that have been allocated in its context during [its] life" will be released, and also "any context of this scope instance" will be "cleaned up". This passage is, however, insufficient to specify which contexts these are. In particular it does not imply that the context copied for remote method invocation is released or cleaned up

at the same time due to they being associated with the same scope instance.

6. In summary, the board finds that the two amendments mentioned under point 3 above go beyond the application as originally filed and, therefore, do not comply with the requirements of Article 123(2) EPC.

7. There being only one sole request, the appeal thus has to be dismissed.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

A. Chavinier-Tomsic Martin Müller

Decision electronically authenticated