**Internal distribution code:**

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

# Datasheet for the decision of 11 March 2016

| | |
|---|---|
| **Case Number:** | T 1370/11 - 3.5.06 |
| **Application Number:** | 03014099.0 |
| **Publication Number:** | 1383045 |
| **IPC:** | G06F9/54 |
| **Language of the proceedings:** | EN |

**Title of invention:**
Extensible on-demand property system

**Applicant:**
Microsoft Technology Licensing, LLC

**Headword:**
On-demand property system/MICROSOFT

**Relevant legal provisions:**
EPC 1973 Art. 56

**Keyword:**
Reduced execution time of a computer-implemented method - not in general a technical effect
Inventive step - technical contribution (no)

**Decisions cited:**
G 0003/08, T 1173/97, T 0049/99, T 0641/00, T 2048/07,
T 0042/10

**Catchword:**
See reasons 10. to 10.5

European Patent Office
D-80298 MUNICH
GERMANY
Tel. +49 (0) 89 2399-0
Fax +49 (0) 89
2399-4465

**Case Number: T 1370/11 - 3.5.06**

# D E C I S I O N
## of Technical Board of Appeal 3.5.06
## of 11 March 2016

|  |  |
|---|---|
| **Appellant:** (Applicant) | Microsoft Technology Licensing, LLC<br>One Microsoft Way<br>Redmond, WA 98052 (US) |
| **Representative:** | Grünecker Patent- und Rechtsanwälte<br>PartG mbB<br>Leopoldstraße 4<br>80802 München (DE) |
| **Decision under appeal:** | **Decision of the Examining Division of the European Patent Office posted on 21 December 2010 refusing European patent application No. 03014099.0 pursuant to Article 97(2) EPC.** |

## Summary of Facts and Submissions

I. The appeal lies against the decision of the examining division, dispatched on 21 December 2010, to refuse European patent application No. 03 014 099.0 on the basis that the main request lacked inventive step, Article 56 EPC, over document

D2: Jorgensen B. N. *et al.*: "Classifying Component Interaction in Product-line Architectures"; TOOLS Pacific 1999, 32nd Int. Conf. on Technology of Object-Oriented Languages and Systems, Melbourne, Australia; 22-25 November 1999; IEEE; pages 66-77.

The sole auxiliary request was not admitted pursuant to Rule 137(3) EPC.

II. A notice of appeal was received on 11 February 2011, the appeal fee being paid on the same day. A statement of the grounds of appeal was received on 31 March 2011 along with two sets of claims 1 to 10 according to a main request and an auxiliary request. The appellant requested that the decision under appeal be set aside and that a patent be granted on the basis of either of the two sets of claims and the following application documents:

description, pages
1, 3 to 14 and 16, as originally filed;
2, 2a, 2b and 15, as received on 10 December 2007.

figures, sheets
1 to 7, as originally filed.

III. Claim 1 according to the main request reads as follows:

"A computer-implemented method for managing properties (216, 218, 232, 234, 236, 238, 240) of objects (404), the method comprising:

associating an expression (212, 214, 222, 224, 226, 228, 230) with a property of an object, the expression defining a relationship between the property and at least one source property;

in response to a request for a value of the property, checking (502) a cache to determine whether a value of the property has been previously cached, and if not, evaluating (506) the expression to determine the value of the property, returning the value of the property and updating (512) the cache; and

invalidating (608, 706) the value of the property if the at least one source property changes, wherein invalidating comprises clearing (706) the value of the property from the cache."

Claim 1 according to the auxiliary request reads as follows:

"A computer-implemented method for managing properties (216, 218, 232, 234, 236, 238, 240) of objects (404), the properties being registered and pertinent information regarding the registered properties is stored in a property entry for each property, the property entry for each property having a property identifier field (436) storing a unique property identifier and

an expression field (438) storing an expression object associated with the unique property identifier, the expression object being created from an expression class providing an evaluation mechanism through a method (304) for evaluating properties and a dependency mechanism through a method (306) for describing the relationship between properties, comprising:

invoking the method for evaluating properties to determine the value of a property, and returning the value of the property; and

invoking the method for describing the relationship to invalidate (608) every inherited property on every child object if a parent property changes."

Both requests also comprise an independent claim for a computer-readable medium and one for a computer system, which correspond closely to the respective independent method claim.

IV. In a communication annexed to the summons to oral proceedings, the board informed the appellant of its preliminary opinion that the claimed invention lacked inventive step over D2, and also because it was confined to matters of programming and programming languages and thus did not have any technical effect.

V. Oral proceedings were held on 11 March 2016, at the end of which the chairman announced the decision of the board.

## Reasons for the Decision

*Article 11 RPBA*

1. The appellant argued that the examining division was wrong not to admit the auxiliary request (see grounds of appeal, page 3, penultimate paragraph) but did not contend that this constituted a fundamental deficiency of the first-instance proceedings in the sense of Article 11 RPBA or a substantial procedural violation in the sense of Rule 103(1)(a) EPC.

1.1 The board notes that the auxiliary request filed with the grounds of appeal is not identical to the one not admitted by the examining division (see annex 5 of the minutes). In other words, the appellant did not maintain the auxiliary request which it argued should have been admitted by the examining division.

1.2 According to the minutes, the appellant filed several new requests during the oral proceedings, each replacing an earlier request. All but the last one were admitted by and discussed with the examining division (see the minutes, points 5, 7, 10, 13 and 15). The board cannot find anything to object to in the conduct of the examining division.

1.3 The board thus saw no reason to remit the case to the department of first instance for further prosecution under Article 11 RPBA without assessing its merits.

*The invention*

2. In general, the application refers to software applications written in terms of "objects" and their "properties".

2.1 When the value of one "first" property depends on that of another "second" one, the question arises of how to update the first if the second changes. The application discloses that the prior art uses so-called "listener functions" which detect changes of the second property and, when that happens, change the first property accordingly (see the description, page 1, lines 1-25).

2.2 It is mentioned that programmers may unintentionally define circular dependencies between properties. When listener functions are used, such circular dependencies may lead to infinite computations because the properties involved in a circle may continue to change states indefinitely (see page 2, paragraph 1). It is also suggested that the storage requirements of the listener functions are undesirable (page 1, last paragraph).

2.3 In order to overcome the shortcomings of the prior art, the invention proposes an "on-demand type of property system" (see page 2, last paragraph, line 4, and page 4, lines 4-5), i.e. a system in which "a change in a [...] property does not affect a dependent property until an operation, associated with the dependent property, is called" (page 4, lines 5-7).

2.4 Figure 5 shows the proposed process of retrieving the value of a property. First a "cache" is checked to determine whether a value for the property has been cached (502 and 504 in figure 5; page 12, lines 13 to 18). If so, the cached value is used (page 12, lines 18 to 19). If not, an expression is evaluated, and the resultant value is returned and used to update the cache (506, 508 and 512 in figure 5; page 12, line 21 to page 13, line 4).

2.5 If the value of a property is changed, the process depicted in figure 6 ensues. The new value of a changed property is stored (606 in figure 6; page 14, lines 1 to 5) and the dependent properties are notified of the change in order to invalidate the "dependents" (608 in figure 6; figure 7; page 14, lines 6 to 27).

2.6 It is disclosed that invalidation is recursive (page 14, lines 9 to 11) and that circular dependencies may occur (page 2, lines 1 to 2, and page 15, lines 1 to 6). However, the invention allows such circularity to be detected if the "property engine maintains a list of properties which have been invalidated", so that infinite computation can be avoided (*loc. cit.*).

*The prior art*

3. D2 refers to the architecture of a software system composed of so-called "components" which may "depend on other components to fulfill their purpose" (see page 67, paragraph 2). The corresponding communication between components may operate according to a "push" or a "pull" model (or, in fact, a "mixed model"; see page 67, paragraph 3). According to the "push" model, the supplier is responsible for communicating data updates, whereas according to the "pull" model it is the consumer's responsibility to issue data requests when necessary. More specifically, D2 discloses that component "properties" (see page 68, paragraph 3) "can be computed based on other properties" (see page 69, paragraph 2). If "eager evaluation" is used here, the property's "setter method" will generate a "property change event" (i.e. an event will be generated whenever the property changes); if "lazy evaluation" is used, the property's "getter method" will recompute the property value (i.e. only when needed). Eager and lazy evaluation effectively correspond to the "push" and "pull" models.

*Main request*

4. It is common ground between the examining division in its decision and the appellant that D2 constitutes a suitable starting point for the assessment of inventive step. The board agrees.

5. Part of claim 1 relates to terms in which any application using the invention will have to be modelled, namely objects, properties, relationships (i.e.

dependencies) between properties and "expressions" defining how the value of a "source" property determines the values of a dependent property. Claim 1 also refers to the possibility that the value of the source property might change.

5.1 The board considers that document D2 discloses all these features. The "components" correspond to the claimed "objects" and so do the associated "properties". D2 discloses that dependent properties can be computed from others. The skilled person would understand this as a reference to a piece of code to be executed in order to carry out the computation. This piece of code necessarily has to refer to the properties involved and thus implicitly must define the claimed dependency "relationship". Moreover, the board takes the view that this piece of code represents an "expression", broadly construed, to be "evaluated" when computing one property from the other.

5.2 At the same time, the board considers that objects, properties, relationships and expressions are merely abstract concepts used to structure and model an application domain so that it can be implemented on a computer device. The modelling however inevitably precedes the implementation and is, as such, an entirely mental activity which remains within the domain of non-technical knowledge and cannot, by itself, constitute the technical solution to a technical problem. In this, the board endorses the reasoning of T 49/99 (catchword and reasons 7).

6. The remainder of claim 1 specifies that property values may be requested and that they are retrieved from a "cache". When a requested value is missing, the cache is updated to hold the result of evaluating the expression

concerned, and when the source property changes, that value must be "invalidated" and "cleared [...] from the cache".

*Does the invention solve a technical problem?*

7. The appellant argued that the claimed invention solved two technical problems in comparison with D2.

a) The computing time was reduced by evaluating the "expression" only on request - i.e. only when needed - and caching the result, and

b) the problem of circular property dependencies and infinite computations could be avoided.

8. Regarding b), the board agrees that the invention indeed does allow the detection of circular dependencies as disclosed by the application (see the description, page 15, lines 1-6). However, this possibility is a consequence of the fact that property values are computed only on request, regardless of whether the claimed caching method is used or not.

8.1 In this regard, the board notes that computation on request is known from D2, which discloses as "lazy computation" the recomputation of a property value when the "getter method" is called, i.e. when the value is requested (see D2, page 69, paragraph 2).

8.2 Hence, noting that the actual detection of cycles is not claimed, the board finds that the possibility of avoiding infinite computations by cycle detection also exists in D2 for precisely the same reasons as in the invention.

9. Regarding a), the board accepts that the claimed caching method is not known from D2 and has the effect that unnecessary evaluation or re-evaluation of an expression can be avoided.

9.1 It is questionable whether this implies that the overall computing time is actually reduced. That depends on whether the management of the cache is less computationally demanding than the evaluation of the expressions, which in turn depends on the type of expressions and the number of times they typically have to be evaluated. Since, however, nothing is known about the expressions it cannot be demonstrated that the computing time is reduced, either for the general or for the average or typical case.

9.2 However, for the sake of argument, it is assumed that the claimed cache does indeed have the alleged advantage of reducing the computing time required.

*Is the reduction of computing time a technical problem?*

10. In the board's view, the argument that a computer program or computer-implemented method is inventive because it is faster than an earlier one is *on its own* insufficient to establish an inventive step. More specifically, the improved speed of a computer program is not *by itself* a technical contribution to the art.

10.1 By way of illustration, let it be assumed that the method in question is a non-technical one, for example a mathematical method or a method of doing business.

10.2 Under Article 52(2)(a) and (c) and (3) EPC, this method will not be regarded as an invention in the sense of Article 52(1) EPC.

10.3 Any computer program implementing that method will, of necessity, need a particular amount of computing resources, in particular time. This is merely a consequence of the "normal" physical interactions between program (software) and computer (hardware) (see T 1173/97, OJ EPO 1999, 609; headnote and reasons 6.6, 9.6, 13). According to established jurisprudence of the boards of appeal, the computer program would thus be found not to comply with Article 52(2)(c) and (3) EPC for lack of a "further" technical effect. And because the computing time does not contribute to the technical character of the computer program, it could not support the presence of inventive step of a corresponding computer-implemented method (see T 641/00, OJ EPO 2003, 352; headnote 1).

10.4 These findings cannot be changed by a document which discloses an alternative, earlier non-technical method which takes longer to carry out on a computer. In other words, it cannot be argued that a computer program must be regarded as an invention in the sense of Article 52(1) EPC, i.e. as a technical invention, *for the sole reason that* an earlier computer program happens to exist which solves the same, non-technical problem more slowly (see also T 42/10, reasons 2.12, for a similar conclusion). Otherwise, the exclusion of computer-implemented methods under Article 52(2) and (3) EPC would become meaningless, because for any given computer program a less efficient one is either known or conceivable. As a consequence, a computer-implemented method cannot be found to show an inventive step for the sole reason that a slower computer-implemented method exists in the prior art.

10.5 Therefore, for a computer-implemented method or a computer program to be patentable it must be established that it has a "further" technical effect and solves a technical problem independently of its absolute or relative computing time. Only then, and only if the alleged speed-up affects an established technical effect, can it be argued that the speed-up contributes to a technical effect and thus to inventive step (see again T 641/00, headnote 1).

11. Returning to the claimed invention (see point 9.2), the board accepts that certain kinds of cache may well be accepted as the technical solution to a technical problem. This may be the case, for instance, for a fast page cache in RAM to avoid repeated access to a slow peripheral hard disk.

11.1 However, the claimed "cache" is not such a cache. In fact, little detail of the cache is claimed. For example, the claims cover the possibility that each object stores the values of its dependent properties in place along with a flag indicating their validity, and that the term "cache" is merely a way of referring to the totality of these local property copies.

11.2 Thus, the claimed concept of caching does not go beyond the idea of memorising a value that one might need again and its validity, so that the effort involved in obtaining it need not always be repeated. In the board's view, this idea is obvious from first principles.

11.3 Moreover, the claimed method of "caching" a value and its validity achieves, if anything, a speed-up of an entirely abstract method (see point 5 above) and thus is, as argued above, insufficient to establish that the claimed computer-implemented method solves a technical problem.

12. In summary, the appellant did not convince the board that the claimed invention had any genuinely technical effect over D2. In fact, it also failed to show that the claimed method could be considered to have any "further" technical effect beyond being computer-implemented.

13. Therefore, the board finds that the subject-matter of claim 1 according to the main request lacks inventive step, Article 56 EPC 1973.

Auxiliary request

14. The independent claims of the auxiliary request differ from those of the main request in that they specify a number of details regarding the implementation of objects, properties and expressions.

14.1 In particular, they indicate that for each property of an object a "property identifier field" and an associated "expression field" are provided, the latter storing an "expression object" created from an "expression class" and providing two methods, one for evaluating the property based on the source property and one for invalidating dependent properties based on the separately defined "relationships".

14.2 In the board’s judgement, these features are merely particular choices a computer programmer might make when implementing the method according to claim 1 of the main request, namely choosing an object-oriented language for this purpose and coding the specific functions required in terms of the programming language chosen. Choices of this type form part of the art of computer programming and cannot in themselves be adduced as an indication of "technical" activity (see T 2048/07, reasons 7.3, and G 3/08, OJ EPO 2011, 10, Reasons 13.5).

14.3 Therefore, the board also finds that the subject-matter of claim 1 according to the auxiliary request lacks inventive step, Article 56 EPC 1973.

## Order

**For these reasons it is decided that:**

The appeal is dismissed.

The Registrar: The Chairman:

Beschwerdekammern des Europäischen Patentamts
Boards of Appeal of the European Patent Office
Chambres de recours de l'Office européen des brevets

B. Atienza Vivancos W. Sekretaruk

Decision electronically authenticated