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

Datasheet for the decision of 15 July 2010

Case Number:	T 1133/07 - 3.5.01
Application Number:	01301591.2
Publication Number:	1146442
IPC:	G06F 17/30
Language of the proceedings:	EN

Title of invention:
Method and apparatus for storing and accessing electronic content

Applicant:
Fujitsu Services Limited

Opponent:
-

Headword:
Accessing objects/FUJITSU

Relevant legal provisions:
-

Relevant legal provisions (EPC 1973):
EPC Art. 56

Keyword:
"Inventive step - choosing object version to modify (no)"
"Inventive step - marking object as deleted (no)"

Decisions cited:
-

Catchword:
-

Case Number: T 1133/07 **-** 3.5.01

D E C I S I O N
of the Technical Board of Appeal 3.5.01
of 15 July 2010

Appellant:	Fujitsu Services Limited 26 Finsbury Square London, EC2A 1SL (GB)
Representative:	Brandon, Paul Laurence APPLEYARD LEES 15 Clare Road Halifax HX1 2HY (GB)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted 1 February 2007 refusing European patent application No. 01301591.2 pursuant to Article 97(1) EPC 1973.**

Summary of Facts and Submissions

I. This appeal is against the decision of the examining division to refuse the European patent application No. 01301591.2 according to the state of the file on the grounds that claim 1 of the sole request was not clear (Article 84 EPC 1973) and was not new (Article 54(2) EPC 1973) over D3 - W. Wieczerzycki: "Versioning Technique for Collaborative Writing Tools", 7th International Workshop on Database and Expert Systems Applications (DEXA'96), Zurich, Switzerland, 9-10 Sept 1996, Los Alamitos, CA, USA, IEEE Comput. Soc, US, pages 463-468.

II. In the "Introduction" (and essentially also at point 3.2 under "First Auxiliary Request") of the statement setting out the grounds of appeal, dated 11 June 2007, the appellant stated that the appeal was based on the main request filed therewith. However, the main request was missing from the documents filed. Under the heading "Main Request", it was requested that "the application be allowed in its present form". The appellant filed a first to fourth auxiliary request and an auxiliary request for oral proceedings.

III. In the communication accompanying the summons to oral proceedings the Board assumed that the main request was the most recently amended set of claims, i.e. claims 1 to 9, filed with the letter of 20 March 2006 as mentioned in the examining division's communication of 26 October 2006, forming the basis of the decision. The Board summarised the issues to be discussed and tended to agree with the examining division that the subject-matter of claim 1 was neither clear nor novel. The

Board also tended to consider that the subject-matter of the auxiliary requests did not involve an inventive step.

IV. In a reply, it was stated that the applicant would not be represented at the oral proceedings.

V. At the oral proceedings, which took place in the appellant's absence, the Board verified the appellant's requests as above, namely that the decision under appeal be set aside and that a patent be granted on the basis of the main request (claims 1 to 9) filed with letter dated 20 March 2006 before the examining division and underlying the impugned decision, or on the basis of auxiliary requests 1 to 4 filed with the statement of grounds of appeal, dated 11 June 2007. At the end of the oral proceedings, the Chairman announced the decision.

VI. Claim 1 of the main request reads as follows:

"A computer-implemented method for storing and accessing objects in an electronic content store, the method including assigning each object a version number, such that multiple instances of the same object with different version numbers may coexist; wherein instances of objects may be accessed for reading or writing by specifying a requested version number; characterised in that
(a) when reading an object, if an instance of that object with the requested version number exists in the content store, the method returns that instance of the object, and otherwise the method returns the most recent older instance of that object; and

(b) when writing to an object, if an instance of that object with the requested version number exists in the content store, the method performs the write to that instance of the object, and otherwise the method makes a copy of the most recent older instance of the object, assigns the requested version number to the copy, and performs the write to the copy."

Claim 1 of the first auxiliary request adds to the end of claim 1 of the main request the feature "the objects are independently addressable".

Claim 1 of the third auxiliary request adds to the end of claim 1 of the main request the feature that "an object is deleted at a specified version number by writing to the content store an instance of the object with the specified version number and marked as deleted, this instance replacing any existing instance of the object with the same version number".

Claim 1 of the second and fourth auxiliary request essentially specifies in claim 1 of the first and third auxiliary request, respectively, that the "content store access service" rather than the method modifies the objects.

Reasons for the Decision

1. The appeal complies with the requirements referred to in Rule 65(1) EPC 1973 and is therefore admissible.

The application

2. The application relates to storing and accessing data used in websites. The pages of modern websites are made up from a variety of different types of content. The simplest types are text and images that are present in nearly all pages (paragraph 2 of the published application). There might also be multimedia files (audio and video - paragraph 2). If the pages relate to interactive services, there may be forms for accessing data in a database. To give a website a consistent appearance pages are usually based on templates that must also be stored (paragraphs 3 and 21). A program ("template renderer") fills in these templates with static data (e.g. text) or dynamic data (e.g. data from a form) to produce the final pages (paragraph 36). A template may be only partially filled in and stored for later completion ("partially-rendered template" - paragraph 39).

3. The invention stores all this content as a set of objects in a content store arranged as a hierarchy or tree structure (paragraph 11, Figure 2). This hierarchy reflects how the pages are organised (e.g. the /sport object contains all the information about sport, including objects that relate to news and articles). Each object has an address like a file in a directory file system (paragraph 12) and a number of properties (paragraph 13).

4. A key idea of the invention is that one of the properties of an object can be a "temporal version number", which may relate to a publication date or a development phase (paragraph 16). This allows the

content store to store different versions of a given object with the same address but different version numbers. An object is thus accessed by its address and version number.

Main request

5. Claim 1 of the main request specifies that when the data is accessed the following rules must be observed:

(a) when reading an object, if an instance of that object with the requested version number exists in the content store, the method returns that instance of the object, and otherwise the method returns the most recent older instance of that object; and

(b) when writing to an object, if an instance of that object with the requested version number exists in the content store, the method performs the write to that instance of the object, and otherwise the method makes a copy of the most recent older instance of the object, assigns the requested version number to the copy, and performs the write to the copy.

6. In examination proceedings, the applicant explained how the read rule operates in connection with Figure 3. Thus, for example, object B has two versions: one with version number = 1 (unshaded) and another with version number = 4 (shaded). If a read request is made for version 2 or 3 of object B, the system will actually return version 1 (the most recent older instance); the first version is therefore part of system versions 1 - 3. Similarly, if a request is made for version 5 of object B, the system will actually return version 4;

the second version is therefore part of system versions 4 and 5. The Board notes that the above assumes that the version number of an object relates to the development phase. In effect, the read operation returns the object with the required or nearest lower development phase number.

7. The first part of the write rule results in an object with the specified version number being overwritten if it exists. If it does not exist, the rule provides for making a copy of the most recent older instance of the object, assigning the requested version number to the copy, and performing the *write to the copy*. However, in the Board's view, the concept of writing to the copy does not make sense since there is no point in making a copy if it is to be overwritten. Paragraph 17 of the description, however, states that *modifications are applied to this copy*. This means that the write operation is in fact an update that results in a modified version of the most recent older version and is, in the Board's view, the only sensible interpretation of this feature.

8. In connection with the preamble of claim 1, which defines that each object has a version number that is used when accessing the object, the examining division observed in the communication forming the basis of the decision to refuse the application that:

"According to D3, objects, of which documents are composed are stored in a database. It is to be noted that objects in the database can coexist in multiple versions corresponding to different versions of documents (see page 465, right-hand column, first full

paragraph, last paragraph). Thus, assigning versions of objects to different versions of a document, corresponds to assigning version numbers to version instances of objects (compare with the description, page 5, last paragraph [paragraph 16 of published application]). In this regard, it is to be noted that three versions of an object figure_1.2 exist physically in the database: one corresponds to the document DOC_A_1, one corresponding to the document DOC_A_2 and one corresponding to the document DOC_A_3 (see page 465, right-hand column, last paragraph; figure 2).
It is also disclosed in D3, that objects, as disclosed in D3, can be updated and read by users (see page 465, left-hand column, last paragraph; page 466, left-hand column, first paragraph)."

In connection with the preamble of claim 1, the division also cited D3 at page 465, left-hand column, penultimate paragraph, but without any explanation. This paragraph discloses that it is possible to update individual document components (objects). In the Board's view this discloses that each object effectively has a version number via the document version and that the objects can be accessed, in the terms of the preamble.

9. In connection with the read rule (a), the examining division observed that:

"It is to be noted that, according to D3, e.g. DOC_A_2 physically shares most of objects with its parent DOC_A_1 except Subsection_1_2 and Fig_1_2 that were modified after document version derivation (see page 465, right-hand column, last paragraph; figure 2). This policy was chosen in order to avoid redundant storage of objects. See D3 page 465, right-hand column, first full paragraph; compare with the description, page 6, penultimate paragraph [paragraph 18 of the published application].

"Moreover, when a user makes a request to read DOC_A_2, all objects corresponding to this document are retrieved from the database, as implicitly disclosed in D3. In such a case objects versions corresponding to DOC_A_2 are requested. Since e.g. Subsection_1_1 is shared at the physical level with DOC_A_1, its version corresponding to DOC_A_1 must be retrieved (the most recent older instance), as implicitly disclosed in D3. In turn, Subsection_1_2 was modified and the version of this object, that corresponds to the document DOC_A_2, exists at the physical level (the requested version instance exists physically) and may be retrieved in response to the users request, as implicitly disclosed in D3. Compare with page 465, left-hand column, last paragraph - page 465, right-hand column, second paragraph."

The Board agrees with the examining division that D3 discloses the read rule.

10. In connection with writing to an existing version of an object according to the first part of write rule (b), the examining division observed that:

"When a user wishes to perform a non-versioning update of (to write to) the object Subsection_1_2 in DOC_A_2, the system perform writes to the version instance of the Subsection_1_2 that corresponds to the DOC_A_2, for this version instance exists at the physical level (the object with a requested version instance is updated)."

The Board agrees that, in D3, a non-versioning update of an existing object in a document overwrites that object and thus anticipates the first part of rule (b).

11. For non-existing versions the division said:

"When, however, the non-versioning update should be carried out on the Subsection_1_1 of the document DOC_A_2, firstly the instance of Subsection_1_1 corresponding to DOC_A_1 is found by the DBMS and retrieved to the memory (the most recent older instance is copied to the memory), as implicitly disclosed in D3. Then the copy is modified and stored physically in the database as the version instance of Subsection_1_1 that corresponds to DOC_A_2 (the requested version number is assigned to the newly created instance of Subsection_1_1), as implicitly disclosed in D3."

Thus, the examining division considered that it was implicit in D3 that if an object was to be written to that existed physically only in the parent document, a copy of that version was modified. The Board agrees that the passage in the first paragraph at page 466, left-hand column, states that the DBMS would find the object, but the question is if it were to be modified whether the system would make a copy of it and modify the copy or simply modify the original version. The Board is not convinced that the former is actually implicitly disclosed. However, the Board judges that choosing which versions are to be updated is only a matter of preference, the mere idea of which is not technical and thus cannot contribute to inventive step. Modifying the system to allow this would be a matter of routine design, also not involving an inventive step. In any case, the skilled person would probably try to avoid unexpected effects such as modifying previous versions. Thus, the Board cannot see how the write rule involves any inventive subject-matter.

12. Thus, the Board judges that claim 1 of the main request does not involve an inventive step (Article 56 EPC 1973).

13. The appellant's grounds for overturning the decision are rather terse, consisting at point 2.2 of the grounds solely of the statement "in D3 there is no possibility of writing to older versions of an object". However, even if this is not implicitly disclosed by the possibility of updating document components and the ability of the DBMS to access a version of a particular component (discussed above at points 8 and 11, respectively), the Board considers that the mere idea of deciding what to be able to update is a non-technical preference that would not involve an inventive step as discussed above in connection with the write rule.

Auxiliary requests

14. Claim 1 of the first auxiliary request adds the feature that "the objects are independently addressable". The appellant's argument at point 3.4 of the grounds of appeal is that in D3 the whole document is derived if a part of it is modified, whereas the invention reduces this computational effort. However, regardless of

whether the whole document is derived when an object is updated (and this is only in the case of a so-called *versioning update)*, the user can cause the DBMS to find any version of an object as discussed above at point 11. In the Board's view, this means that the objects are "independently addressable". Thus, the Board judges that claim 1 of the first auxiliary request does not overcome the above objections and therefore also does not involve an inventive step.

15. Claim 1 of the third auxiliary request essentially adds the feature of originally filed claim 3, relating to the deletion of an object. This feature boils down essentially to specifying that an object is deleted by simply marking the relevant version as deleted. However, the Board judges that actually deleting or marking something as deleted are very well known alternatives in the present field, so that this choice can not involve an inventive step. The use of the "nil" designation for objects that do not exist in a given document version, as described in D3 at page 465, right-hand column, second full paragraph is an example of such a technique.

16. The amendments to independent claim 1 of the second and fourth auxiliary requests only relate to the clarity objection that the examining division raised about which unit performs the steps of the method. They do not overcome the above objections, so that claim 1 of the respective requests does not involve an inventive step.

17. Since none of the requests is allowable, it follows that the appeal must be dismissed.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

T. Buschek S. Wibergh