BESCHWERDEKAMMERN DES EUROPÄISCHEN PATENTAMTS

BOARDS OF APPEAL OF THE EUROPEAN PATENT OFFICE

CHAMBRES DE RECOURS DE L'OFFICE EUROPEEN DES BREVETS

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

Datasheet for the decision of 4 November 2011

Case Number:	T 1500/08 - 3.5.01
Application Number:	04003750.9
Publication Number:	1566745
IPC:	G06F 17/30
Language of the proceedings:	EN

Title of invention:
Generation of database queries from database independent selection conditions

Applicant:
UBS AG

Headword:
Generation of database queries/UBS

Relevant legal provisions:
EPC Art. 52(1)

Relevant legal provisions (EPC 1973):
EPC Art. 54(1), 56

Keyword:
"Novelty (no): main request"
"Inventive step (no): auxiliary requests 1 and 2"

Decisions cited:
-

Catchword:
-

Case Number: T 1500/08 - 3.5.01

D E C I S I O N
of the Technical Board of Appeal 3.5.01
of 4 November 2011

Appellant:	UBS AG Bahnhofstrasse 45 8001 Zürich (CH)
Representative:	Röthinger, Rainer Wuesthoff & Wuesthoff Patent- und Rechtsanwälte Schweigerstrasse 2 81541 München (DE)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted 17 December 2007 refusing European patent application No. 04003750.9 pursuant to Article 97(1) EPC 1973.**

Composition of the Board:

Chairman:	S. Wibergh
Members:	P. Scriven
	P. Schmitz

Summary of Facts and Submissions

I. The appeal is against the Examining Division's decision to refuse European patent application 04003750.9. The refusal was on the grounds that the invention was obvious to the skilled person, on the basis of the teaching of document D3 (Williams and Lane, "Web Database Applications with PHP and MySQL", O'Reilly, 2002). The principal argument was that D3 disclosed the use of query language statements with placeholders for selection values, and that the use of similar placeholders for selection operators was obvious.

II. With the statement setting out the grounds of appeal, the appellant requested that the Examining Division's decision be set aside, and that a patent be granted on the basis of a main request, or else on the basis of one of two auxiliary requests. The appellant also requested oral proceedings, if the main request were not allowed.

The appellant argued that the Examining Division had erred in finding the use of placeholders for selection operators obvious. The skilled person would have had no reason to consider selection operators at all, because there was no motivation to do so in any of the cited prior art. In particular, the prior art did not teach the use of placeholders for selection operators.

III. The Board scheduled oral proceedings. With the summons, the Board sent a communication setting out its provisional view. In particular, the Board drew attention to D2 (US-B 6285998), which disclosed a method allowing users to choose both selection values and selection operators.

IV. With its reply to the summons, the appellant filed a new auxiliary request 2, and presented arguments in favour of novelty and inventive step. In particular, the appellant thought that D2 was the proper starting point for the assessment of inventive step, and that claim 1 according to auxiliary request 2 defined several novel features which would not have been obvious to the skilled person.

V. During the oral proceedings, held on 29 September 2011, the appellant filed amended versions of its three requests, and requested that the decision under appeal be set aside and that a patent be granted on the basis of the main request, or, alternatively, on the basis of auxiliary request 1 (claim set I) or auxiliary request 2 (claim set II), as amended during oral proceedings.

Claim 1 according to the **main request** reads as follows.

A method of generating a database query from a database independent search request, comprising the steps performed by a computer component (14)

– providing in a storage device (23) or via access to a storage device (23) one or more predefined query strings in a database specific query language format, wherein the predefined query strings include one or more first place-holders (#) that are substitutes for database specific selection operators;

- receiving a search request in a database independent format, the search request specifying at least one selection condition;
- deriving one or more database specific selection operators from the at least one selection condition specified in the search request; and
- substituting a first placeholder (#) in one or more of the predefined query strings with at least one previously derived database specific selection operator when generating the database query.

Claim 1 according to **auxiliary request 1** is identical, except that "wherein the predefined query strings include one or more first place-holders (#)" reads "wherein the predefined query strings include multiple identical first place-holders (#)".

Claim 1 according to **auxiliary request 2** reads as follows.

A method of generating a database query from a database independent search request, comprising the steps performed by a computer component (14)

- providing in a storage device (23) or via access to a storage device (23) one or more predefined query strings in a database specific query language format, wherein the predefined query strings include one or more first place-holders (#) that are substitutes for database specific selection operators and one or more second placeholders (@xyz) that are substitutes for selection values;

- receiving a search request in a database independent format, the search request specifying at least one selection condition;
- deriving, on an application layer (14), one or more database specific selection operators from the at least one selection condition specified in the search request; and one or more database specific selection values (@abc) from the search request taking into account respective selection operators specified in the search request;
- substituting, on the application layer (14), a first placeholder (#) in one or more of the predefined query strings with at least one previously derived database specific selection operator when generating the database query, wherein the second placeholders that substitute the selection values remain in the database query;
- transferring the selection values and the database query with substituted first placeholders (#) but non-substituted second placeholders (@xyz) to a database server (18); and
- substituting, by the database server (18), the second placeholders with the appropriate selection values when generating the final database query.

VI. The appellant argued that D2 failed to disclose pre-defined strings in the query language, and, therefore, the use of placeholders in such strings; or the separation between the application layer and the database server. Under some circumstances, these features resulted in a more efficient method, but the method disclosed in D2 was not compatible with either of them. In addition, the appellant argued that D1 (US-A 6105043), and D3 directly contradicted the teachings of D2, so that the skilled person could not combine their teachings with that of D2. Furthermore, since neither D1 nor D3 disclosed the use of placeholders for selection operators as well as for selection values, they could not teach the dependence of selection values on the operators, as defined in claim 1 according to auxiliary request 2.

VII. At the end of the oral proceedings, the chairman announced that the decision would be issued in writing.

Reasons for the Decision

1. Background

1.1 The invention concerns the generation of database queries. Such queries need to be presented to a database server in a specific format. SQL is an example of a language which imposes such formatting. Formal queries are difficult to write, and so there are programs which present the user with a relatively intuitive interface, and convert her input to a formal statement in the query language. The conversion between user input and query language statement is where the invention comes in.

1.2 It was known, at the filing date, to maintain a number of query strings, with placeholders for selection values. That has been acknowledged by the appellant, and D3 is an example. A typical fragment of a query string might be *WHERE date = @yyyy*. After the user has defined a search request using the interface, the value of *@yyyy* is extracted and substituted for *@yyyy* in the query string, to produce *WHERE date = 2010*, for example. Thus, *@yyyy* is simply there to mark the place in which the value the user enters will be put. It is what the application terms a *placeholder*.

1.3 The invention, in its basic form, uses placeholders for selection operators. An example of the sort of query string involved would be *WHERE date # 2010*. Now, when the user has defined a search request, the placeholder *#* will be replaced by an operator, for example =, < or <=, so that the final query language fragment might be *WHERE date = 2010* or *WHERE date < 2010* or *WHERE date <= 2010*.

2. Document D2

2.1 D2 discloses a method which allows a user to define a search, and produces the corresponding statement in a query language. That is, broadly, the sort of thing outlined under 1.2.

2.2 D2 starts from, and distinguishes itself from, its own prior art, which it sets out in column 1. That prior art can be termed the "parameterised query method". It consists of storing query language statements, in which some terms are "parameters". When the query is executed, the database server detects the presence of parameters, and prompts the user to provide values. Once the answer is received, the query is executed, with the values in place of the parameters. In the parlance of the present application, the parameters are placeholders.

2.3 D2 identifies some problems with that (column 1, lines 55 - 66). It is slow because the database server has to wait for the user, at the client, to respond. It is also not easy to use.

2.4 D2 solves these problems by providing a sequence of dialogue boxes, which allow the user to define a search using ordinary English statements, and turns that into a query which is sent to the database server. The user can define some terms of the search to be "'ask me' values". If she does that, then, when the search is executed, she will be prompted to provide values; but those values are put into a query string before it is sent to the database server.

2.5 What makes D2 particularly relevant to the present case is that not only selection values and field names, but also selection operators can be "ask me" values (e.g. column 5, lines 56 - 61).

2.6 The details of the disclosure of D2, and how they relate to particular features defined in the claims, are discussed below.

3. The main request

3.1 The terms *database specific*, and *database independent*, as used in claim 1, need some explanation. The applicant's intention, as explained during oral proceedings, is that the user can enter a search request in a format which is not in the query language understood by the database server, and that the *predefined query strings* are strings in the format that the database server does understand (e.g. SQL). The claim is not completely clear on that point, but the

Board, in reaching its decision, has followed the appellant's interpretation.

3.2 On that understanding, the Board sees D2 as disclosing the whole subject matter of claim 1.

3.3 Claim 1 defines the provision of (access to) stored query strings, which are in the format of the query language, and which have at least one placeholder for a selection operator. D2 certainly discloses the storage of queries with placeholders ("ask me" values) which may be for operators (D2, column 5, lines 56 - 61). The appellant, however, denied that those queries are in the format of the query language.

3.4 D2 does not clearly identify how the user's queries, with "ask me" values, are stored. There is nothing to suggest they are necessarily in the form of a query language statement (see column 6, lines 1 - 17).

3.5 The Board concludes that the "ask me" values are not disclosed as placeholders within a string of the query language.

3.6 When a search is performed, the method of D2 prompts the user for the "ask me" values. They, together with the values already known, are *inserted into a query string* (column 6, line 52). The straightforward reading is that the query string exists, and that values are inserted in the right places. That implies those places are identified; that is, there are placeholders. The appellant, however, asserted, during oral proceedings, that "inserted" could be understood in the sense of "added", and that the skilled person would do something different when implementing D2. He would build up the query string *WHERE date = 2010* in stages as *WHERE*, then *WHERE date*, then WHERE date =, and finally *WHERE date = 2010*. That is a procedure which does not use placeholders. The Board, however, can see nothing in D2 which would suggest to the skilled reader that his straightforward reading was wrong, or would point towards the procedure set out by the appellant.

3.7 The Board's conclusion is that D2 implicitly discloses the use of placeholders in pre-existing query strings. Since "ask me" values can be for selection operators, these placeholders can be too.

3.8 The Board is also satisfied that these query strings are in a query language. The appellant argued against that, because they are not in SQL format. That is based on D2, column 6, lines 64 - 67: *Query parser 207 in server computer 202 parses this compact query string to create a query string suitable for execution by the query engine 208, such as a SQL standard database query*. That argument fails, firstly, because the translation only applies when the query string is sent in what D2 calls a "compact" form, and, secondly, because however it is sent, the string is a form the database server understands. It is the Board's view that that is sufficient for the string to belong to a "query language", as the term is used in the application (paragraph 6 of the published application): *a search request … has to be translated into a database specific query language that can be understood by the database server*.

3.9 It is straightforward to see that D2 also discloses the receipt (from a user) of a search request which is not in a query language, the derivation of selection operators in the query language, and the substitution of placeholders. The appellant made no arguments on those points.

3.10 The Board concludes that D2 discloses the whole combination of features defined in claim 1, and, therefore, that the main request cannot be allowed due to a lack of novelty (Article 54(1) EPC 1973).

4. Auxiliary request 1

4.1 According to the claim 1 of the main request, the predefined query strings include one or more first placeholders; in claim 1 of auxiliary request 1, they include multiple identical first placeholders.

4.2 That is an unclear formulation, but during oral proceedings, the appellant explained its intention as being that the same placeholder, for example the character "#", appears at more than one point in the collection of predefined query strings, but can be replaced by different operators at different places. As an example, a string such as *year # $date AND name # $name* could become *year < 2010 AND name = Smith*.

4.3 The "ask me" values in D2 are used in that way, but they are not in the query language format, as explained at 3.4 and 3.5 above.

4.4 The query language strings into which field names, values or operators specified by the user are inserted

(D2, column 6, lines 49 - 52), while having placeholders, do not necessarily have identical placeholders. D2 says nothing about that. The use of identical placeholders ("ask me" values) in the query as presented to the user would, however, make it obvious to do the same for the placeholders in the query strings.

4.5 The Board concludes, therefore, that the subject matter of claim 1 does not involve an inventive step (Article 56 EPC 1973), and that auxiliary request 1 cannot be allowed.

5. Auxiliary request 2

5.1 Claim 1, as compared to that of the main request, introduces some additional features.

There are now two types of placeholder. One is for selection operators, the other for selection values.

In addition to the derivation of selection operators, selection values are now derived.

The derivation of values and operators is performed *on an application layer*.

The derivation of the selection values is done *taking into account respective selection operators*.

Selection operators are substituted before transmission of the query to the database server; selection values are left to be substituted in the database server.

5.2 That D2 discloses the two types of placeholder is not disputed. That they are in query language format is disputed, but that point has been dealt with (see 3.8).

5.3 That D2 discloses the derivation of selection operators and values in the query language is also not disputed. It is inherent to D2 that whatever values or operators the user specifies have to be turned into query language values and operators.

5.4 The specification that the derivation is performed *on the application layer* is somewhat unclear, especially in combination with other operations on the database server (as opposed to the database layer). As the appellant explained during oral proceedings, the point is that the derivation is performed by the application server, and it is with that interpretation that the Board has reached its decision.

5.5 According to D2, the derivation of operators and values is performed prior to any transmission to the database server. The Board, therefore, understands that it happens in the application server, that is, in the computer running the application which allows users to define and run searches.

5.6 The appellant explained the derivation of selection values *taking into account respective selection operators* by pointing to Example 2 (application as published, paragraph 68) in which the user enters a value "Hans" and the operator "contains". The value "Hans" becomes "%Hans%", by the addition of two truncation characters. The only examples of this sort of derivation involve such truncation characters. It is the Board's view that this is a matter of taking account of how the query language implements different operators. Here, the query language requires the "%" signs, and the skilled person could hardly do anything else than what the query language demands. Thus, although D2 does not say anything about this point, the Board cannot see how this feature could contribute to inventive step.

5.7 According to D2, all substitutions are made before the query is sent to the database server.

5.8 The question of inventive step, therefore, comes down to this: would it have been obvious to the skilled person to modify the teaching of D2 so that only the operators were substituted before transmission to the database server?

5.9 The technical effect is that, sometimes, a previous search can be reused by the database server (application as published, paragraph 0023). The appellant explained, during oral proceedings, that the database server, when processing searches in which only selection values (but not fields or operators) change, can re-use some of the previous results.

5.10 The Board notes, that the application does not explain what the skilled person needs to do in order to make the database server re-use results in this way. In so far as this feature is disclosed in the application, it is simply something that database servers are able to do. The Board considers it something the skilled person would know about.

5.11 The Board also finds the application unclear as to why the re-use should depend on the presence of placeholders in the query it receives. If the database server receives *date < 2011* followed by *date < 2010*, it is not clear that the server is in any worse position regarding re-use, than if it had received *date < @date, @date=2011* followed by *date < @date, @date=2010*, as the application envisages (application as published, paragraph 96).

5.12 It is, then, somewhat doubtful that the technical effect the appellant claims is actually obtained. However, even if the effect is accepted, the Board considers that the invention would have been obvious to the skilled person.

5.13 By assumption, the technical effect is obtained due to the presence of placeholders. The skilled person was aware of the behaviour of database servers regarding re-use, and would have recognised that the re-use obtained in the prior art mentioned in D2 (column 1, from line 41) is lost, in the invention according to D2, because the query strings the database server receives do not have placeholders. That would have been a clear motivation to keep placeholders, if possible. He was also aware (see points 2.3 and 2.4) that the goals of D2 were to make the definition of the search easy for the user, and to avoid the database server having to wait for the user to respond. He would have recognised that D2 achieved the second of these goals by collecting the values and operators needed before sending them to the database server, rather than in the details of how the values collected were transferred to the database server.

5.14 The Board concludes that the skilled person, aware of the way database servers re-use queries, and of the way in which D2 solved the problems it set for itself, would have arrived at the invention defined in claim 1. That is, the Board considers the subject matter of claim 1 to lack inventive step (Article 56 EPC 1973), and cannot allow the request.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar:	The Chairman:
T. Buschek	S. Wibergh