Internal distribution code:

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

Datasheet for the decision of 28 June 2017

Case Number: T 1897/12 - 3.5.07

Application Number: 04782033.7

Publication Number: 1661046

IPC: G06F17/30

Language of the proceedings: EN

Title of invention:
In-place evolution of XML schemas in databases

Applicant:
Oracle International Corporation

Headword:
Schema evolution/ORACLE INTERNATIONAL

Relevant legal provisions:
EPC Art. 56

Keyword:
Inventive step - all requests (no)

This datasheet is not part of the Decision.
It can be changed at any time and without notice.

Europäisches Patentamt
European Patent Office
Office européen des brevets

Beschwerdekammern
Boards of Appeal
Chambres de recours

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

Case Number: T 1897/12 - 3.5.07

D E C I S I O N
of Technical Board of Appeal 3.5.07
of 28 June 2017

Appellant: (Applicant)	Oracle International Corporation 500 Oracle Parkway Redwood Shores, CA 94065 (US)
Representative:	Dendorfer, Claus Dendorfer & Herrmann Patentanwälte Partnerschaft mbB Bayerstrasse 3 80335 München (DE)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 11 January 2012 refusing European patent application No. 04782033.7 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairman	R. Moufang
Members:	R. de Man
	P. Scriven

Summary of Facts and Submissions

I. The applicant (appellant) appealed against the decision of the Examining Division refusing European patent application No. 04782033.7, published as international application WO 2005/022415 and having a priority date of 25 August 2003.

II. The Examining Division decided that the then main request did not meet the requirements of Articles 83, 84 and 123(2) EPC and that the subject-matter of claim 1 was not new in view of the following document:

D1: US 2003/140308 A1, 24 July 2003.

It also decided that the first, second and third auxiliary requests did not meet the requirements of Articles 83, 84 and 123(2) EPC, and that the subject-matter of claim 1 of each of those requests lacked inventive step in view of document D1.

III. With the statement of grounds of appeal, the appellant submitted main and first auxiliary requests, which were amended versions of the second and third auxiliary requests considered in the contested decision.

IV. In a communication accompanying a summons to oral proceedings, the Board introduced the following document:

D7: Laux A.: "Lexus – XML Update Language", September 2000, retrieved from http://www.infozone-group.org/lexusDocs/html/wd-lexus.html.

It expressed the preliminary view that the subject-matter of claim 1 of both requests lacked inventive step in view of document D1.

V. With a letter dated 26 May 2017, the appellant filed amended main and first auxiliary requests.

VI. Oral proceedings were held on 28 June 2017. At the end of the oral proceedings, the chairman pronounced the Board's decision.

VII. The appellant's final requests were that the decision under appeal be set aside and that a patent be granted on the basis of the claims of the main request or, alternatively, on the basis of the claims of the first auxiliary request.

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

"A method of evolving an Extensible Markup Language (XML) Schema, the method comprising:

receiving, at a schema evolver that is executing in a computer system, an XML document that contains elements that each specify an evolution command describing a change to be made to a first XML schema, wherein an evolution command specifies to append a node, insert a node, or delete a node within the first XML schema, wherein the first XML schema is registered in a database, and wherein existing database object types conform to the first XML schema;

based on said first XML schema and said XML document, said schema evolver automatically alters the first XML schema according to the evolution commands specified in the XML document, thus generating a second XML schema;

determining aspects of said existing database object types that do not conform to the second XML schema; and

based on said aspects of said existing database object types that do not conform to the second XML schema, said schema evolver automatically generating one or more first Structured Query Language (SQL) statements that, when executed by a database server, cause the database server to add an attribute to, or delete an attribute from, at least one of said existing database object types,

wherein said first SQL statements, when executed, cause said existing database object types to conform to the second XML schema."

IX. Claim 1 of the first auxiliary request differs from claim 1 of the main request in that the last two paragraphs have been replaced with the following text:

"based on said aspects of said existing database object types that do not conform to the second XML schema, said schema evolver automatically:

(i) generating one or more first Structured Query Language (SQL) statements that, when executed by a database server, cause the database server to add an attribute to, or delete an attribute from, at least one of said existing database object types, wherein said first SQL statements, when executed, cause said existing database object types to conform to the second XML schema, and

(ii) generating one or more second SQL statements that, when executed, cause effects of said

one or more first SQL statements to be reversed."

X. The appellant's arguments as relevant to the decision are discussed in detail below.

Reasons for the Decision

1. The appeal complies with the provisions referred to in Rule 101 EPC and is therefore admissible.

2. *The invention*

2.1 The application relates to "in-place evolution of XML schemas" in the context of relational databases.

Extensible Markup Language (XML) documents specify a tree of data elements and are used to represent hierarchically structured information. The hierarchical structure common to a number of XML documents, including the names and types of their XML elements, may be defined by an XML schema. The background section of the application (paragraph [0016] of the published application) explains that an XML schema may be used to create corresponding database structures, such as a database table with column names and types corresponding to attribute names and types of XML elements in the XML schema.

It is sometimes desirable to modify (or "evolve") an XML schema (paragraphs [0018] and [0019]), for example to add or delete an attribute. When that is done, it is necessary to evolve the corresponding database structures accordingly (paragraph [0020]).

2.2 According to the invention, a description of the changes to be made to an existing XML schema is provided to a "schema evolver" in the form of an XML document specifying a number of "evolution commands". The schema evolver applies the evolution commands to the existing XML schema to obtain an "evolved" XML schema. On the basis of those changes to the XML schema that affect database object types, the schema evolver generates SQL statements that, when executed by a database server, cause the database server to add or delete attributes as required by the evolved XML schema.

3. *Main request - inventive step*

3.1 Document D1 is the publication (before the priority date) of one of the "related" applications listed on pages 1 and 2 of the present application as published. In paragraph [0034], it describes a mechanism for mapping XML schemas to object-relational database systems. The mechanism is implemented by an "XML schema mapper" included in a database server. Upon registration of an XML schema with the database server, the XML schema mapper determines the appropriate database representation for XML documents conforming to the schema, and it generates "mapping information" for mapping between elements of the XML schema and elements of the database representation. According to paragraph [0060], the XML schemas considered in document D1 are themselves XML documents.

3.2 Document D1 explains, in paragraph [0089], that a registered XML schema may be evolved by "re-registering" it with a new (i.e. evolved) XML schema document. This is sufficient if, for example, the change concerns only the names of elements or

attributes and therefore does not affect the underlying database tables (paragraphs [0089] and [0090]). If, on the other hand, the change adds a new element or attribute, the user has to use "ALTER TYPE and/or ALTER TABLE commands to evolve the underlying tables" before modifying and re-registering the XML schema document (paragraph [0091]).

Paragraph [0092] then explains that, in one embodiment, a "1-step XML schema evolution" is provided, in which "a user simply inputs a new XML schema and all underlying type and table alterations are determined implicitly".

3.3 The appellant correctly pointed out that paragraph [0092] relates to a different embodiment than paragraphs [0089] to [0091]. Nevertheless, it has to be read in the context of the paragraphs preceding it. Indeed, paragraphs [0089] to [0092] form a single section with the heading "XML SCHEMA EVOLUTION".

Paragraphs [0089] to [0091] essentially describe a 2-step approach to XML schema evolution, in which the user first "manually" makes any necessary modifications to database structures (by means of the "ALTER TYPE" and "ALTER TABLE" SQL commands) and then "re-registers" the modified XML schema document. The skilled reader of these paragraphs understands that the database structures are to be kept consistent with the registered XML schema: XML schema evolution requires not only an "evolved" XML schema, but also a corresponding modification of the database structures.

Paragraph [0092] discloses an alternative "1-step XML schema evolution" approach, in which the user "simply inputs a new XML schema" and then "all underlying type

and table alterations are determined implicitly". In the Board's view, in this embodiment the system not only automatically determines, but also makes the necessary modifications to the relevant database structures. If the modifications were still to be made manually, the "1-step XML schema evolution" approach would consist of two steps.

3.4 The appellant argued that the wording in paragraph [0092] "all underlying type and table alterations are determined implicitly" implied, at most, the generation of a list of differences, which helped the user in manually determining how the underlying database object types were to be evolved. That alternative interpretation of paragraph [0092] was not inconsistent with paragraphs [0089] to [0091]. The fact that it existed ruled out any implication that the determined alterations were also made. There was implicit disclosure only if, when carrying out the teaching of the cited document, the skilled person inevitably implemented the feature in question.

However, the present issue is not about what would inevitably happen if the skilled person were to carry out the teaching of paragraph [0092]. The question is rather what meaning the skilled person would ascribe to paragraph [0092] in the context of the "XML SCHEMA EVOLUTION" section and of the document as a whole. As explained above, the Board takes the view that the skilled person would, in fact, understand paragraph [0092] as meaning that the automatically determined alterations are also automatically applied.

3.5 Thus, the subject-matter of claim 1 differs from the "1-step XML schema evolution" embodiment of document D1 in the following features:

(i) the schema evolver receives an XML document that contains elements that each specify an evolution command describing a change to be made to the registered XML schema,

(ii) wherein an evolution command "specifies to append a node, insert a node, or delete a node" within the registered XML schema;

(iii) the schema evolver automatically alters the registered XML schema on the basis of the evolution commands, thus generating an evolved XML schema; and

(iv) for the purpose of modifying the database structures so as to conform to the evolved XML schema, one or more SQL statements for adding or deleting attributes to or from database object types are generated.

3.6 Features (i) and (iii) constitute the main difference: instead of re-registering an evolved XML schema as in document D1, the user provides an XML document specifying the individual changes to be made to the registered XML schema in the form of "evolution commands", and the schema evolver applies these changes to the registered XML schema.

The appellant argued that these features addressed the problem of providing an efficient method for evolving an XML schema for a typical use case in which relatively few changes were made. Document D1 was not concerned with that use case and so could not render those features obvious.

3.7 It is true that document D1 does not mention this use case, but that does not prevent the skilled person from considering it. The realisation that an XML schema or

database schema typically evolves incrementally is in itself not inventive. Indeed, even the present application makes no mention of this typical use case in which features (i) and (iii) show their advantage.

In the field of computing, it is well known that an update to a set of electronic data can be communicated either by relaying the set of updated data or by transmitting the set of changes in the data. Where relatively few changes to a large data set are expected, the skilled person understands it to be more efficient to communicate only the changes and to apply them to the existing data set to obtain the updated data set. In view of this common general knowledge, the skilled person, faced with the problem as formulated by the appellant, would arrive at features (i) and (iii), the choice of an XML format for describing the changes being an obvious choice in a context that already uses XML.

3.8 As to feature (iv), in view of the example of adding a new element or attribute given in paragraph [0091] of document D1, it is obvious that a modification of database structures may involve adding or deleting attributes to or from database object types. And given that document D1 explicitly considers mapping XML schemas to SQL database schemas (see e.g. paragraph [0070]), it is obvious to carry out such modifications by means of one or more SQL statements as in the 2-step process.

3.9 Like feature (iv), feature (ii) is rendered obvious by the example of adding a new element or attribute given in paragraph [0091] of document D1.

3.10 For the sake of completeness, the Board adds that document D7 shows that it was in fact known to communicate updates to XML documents in the form of a set of changes also formatted in an XML language. Document D7 describes the "Lexus update language", which is an XML-based language for expressing updates to XML documents. Lexus includes instructions for appending, inserting and deleting nodes (page 2, "Updates" and "Inserts" sections; page 4, "Append" section; page 5, "Remove" section).

3.11 Hence, the subject-matter of claim 1 of the main request lacks inventive step (Article 56 EPC).

3.12 At this point the Board observes that if - contrary to the Board's interpretation - the "1-step XML schema evolution" embodiment of document D1 had required manual application of the determined differences, automating this step would have been an obvious choice, and this further difference would not have changed the outcome.

4. *First auxiliary request - inventive step*

4.1 Claim 1 of the first auxiliary request adds to claim 1 of the main request a step of automatically "generating one or more second SQL statements that, when executed, cause effects of said one or more first SQL statements to be reversed".

Paragraphs [0043] and [0044] of the application explain that the "second" SQL statements are generated to allow a rollback of the database if an error occurs during execution of the "first" SQL statements.

4.2 It is often desirable to perform a sequence of database operations "transactionally", meaning that – conceptually – either the whole sequence of database operations is performed successfully or none of the operations is performed at all. The usual way to implement a transaction is by means of a "rollback" facility: if an error condition occurs during execution of one of the operations, then the effects of the previous operations are undone, i.e. rolled back.

Document D1 gives an example of transactional behaviour in paragraph [0088], which discloses that if registration of an XML schema fails, the database is rolled back to the state before the registration began. This rollback involves dropping any tables and types that were created. And paragraph [0212] describes an embodiment in which registration is performed "in a manner that allows executing compensating action to undo partial effects when errors are encountered during the schema registration process".

As the appellant correctly observed, paragraphs [0088] and [0212] relate to XML schema registration. But paragraph [0088] makes it clear that XML schema registration, like XML schema evolution, may involve the creation of database objects and that such changes may have to be rolled back. In any event, these passages show that transactional behaviour is already contemplated in document D1.

4.3 Thus, in the context of document D1, it would have been obvious to consider the need to roll back changes made to the database in the process of executing a "1-step XML schema evolution". And to carry out such a rollback, which involves undoing the effects of the "first" SQL statements, it would have been obvious to

generate appropriate "second" SQL statements to reverse those effects.

The appellant submitted that the rollback could instead be effected by means of lower-level internal database commands. But the existence of such an alternative does not detract from the obviousness of using SQL statements to carry out actions on an SQL database.

4.4 The subject-matter of claim 1 of the first auxiliary request therefore lacks inventive step (Article 56 EPC).

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

B. ter Heijden R. Moufang

Decision electronically authenticated