

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte MARCUS F. FONTOURA and
VANJA JOSIFOVSLD

Appeal 2007-3225
Application 10/413,244
Technology Center 2100

Decided: February 25, 2008

Before JAMES D. THOMAS, ALLEN R. MACDONALD, and
ST. JOHN COURTENAY III, *Administrative Patent Judges*.

COURTENAY, Administrative Patent Judge.

DECISION ON APPEAL

This is a decision on appeal under 35 U.S.C. § 134(a) from the Examiner's rejection of claims 1-28. We have jurisdiction under 35 U.S.C. § 6(b). We REVERSE. We also enter a new ground of rejection for claims 1-4, 8, 9, 19-22, and 26-28 under the provisions of 37 C.F.R. § 41.50 (b).

THE INVENTION

The disclosed invention relates generally to processing mark-up language data. More particularly, Appellants' invention relates to a single pass system and method for querying streams of XML data (Spec. 1).

Independent claim 1 is illustrative:

1. A system for querying a stream of mark-up language data, comprising:
 - an expression parser that receives a query and generates a parse tree;
 - a system that receives the stream of mark-up language data and generates a stream of events;
 - an evaluator that receives the parse tree and stream of events, and buffers fragments from the stream of events that meet an evaluation criteria; and
 - a tuple constructor that joins fragments to form a set of tuple results that satisfies the query for the stream of mark-up language data.

THE REFERENCES

The Examiner relies upon the following references as evidence in support of the rejections:

Reiner	US 5,742,806	Apr. 21, 1998
Gray	US 5,758,335	May 26, 1998
Jakobsson	US 5,963,932	Oct. 05, 1999
Kircher	US 6,920,462 B2	Jul. 19, 2005
		(filed Nov. 19, 2001)

THE REJECTIONS

1. Independent claims 1, 10, 19, and dependent claim 28 stand rejected under 35 U.S.C. § 102(e) as being anticipated by Kircher.
2. Dependent claims 2-5, 7, 11-14, 16, 20-23, and 25 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Kircher in view of Jakobsson.
3. Dependent claims 6, 15, and 24 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Kircher in view of Jakobsson.
4. Dependent claims 8, 17, and 26 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Kircher in view of Gray.
5. Dependent claims 9, 18, and 27 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Kircher in view of Reiner.

PRINCIPLES OF LAW

Under 35 U.S.C. § 102, “[a] single prior art reference that discloses, either expressly or inherently, each limitation of a claim invalidates that claim by anticipation.” *Perricone v. Medicis Pharm. Corp.*, 432 F.3d 1368, 1375-76 (Fed. Cir. 2005) (citation omitted). To anticipate, every element and limitation of the claimed invention must be found in a single prior art reference, arranged as in the claim. *Karsten Mfg. Corp. v. Cleveland Golf Co.*, 242 F.3d 1376, 1383 (Fed. Cir. 2001); *Scripps Clinic & Research Foundation v. Genentech, Inc.*, 927 F.2d 1565, 1576 (Fed. Cir. 1991).

ISSUE(S)

We decide the question of whether Appellants have shown the Examiner erred in holding that the cited Kircher reference anticipates independent claims 1, 10, and 19. More particularly, we have determined that the following issue is dispositive in this appeal:

Whether Appellants have shown the Examiner erred in finding that Kircher discloses joining fragments “to form a set of tuple results that satisfies the query for the stream of mark-up language data” as claimed (see independent claims 1, 10, and 19).

Appellants contend the Examiner has not shown that the Kircher reference discloses the claimed tuple constructor (App. Br. 7). Appellants state that the claimed tuple constructor (“construction means” in claim 10) *joins* fragments, each of which meets an evaluation criteria, to form a set of tuple results that satisfies the query for the stream of mark-up language data. Appellants contend that Kircher does not disclose joining fragments to form a set of tuple results when querying a stream of mark-up language data (*id.*). To the contrary, Appellants contend that Kircher merely discloses that one or more results, each of which includes *a single value*, can be obtained in response to a query (*id.*). Appellants point to the following sections of Kircher for support: column 3, lines 58-59, discussing reading a single value filter for a condition; column 3, lines 64-67; and column 4, line 66 through column 5, line 2, in both of which “id” is the only attribute returned. Appellants contend that Kircher does not teach that these results can be *joined* to form a set of tuple results nor does Kircher teach that any one of

these results is generated by *joining* fragments (App. Br. 7-8). Contrary to the Examiner's assertion, Appellants contend that satisfying a multiple condition query does not disclose bringing two or more items together to form a unit (App. Br. 9).

The Examiner disagrees. The Examiner notes that Kircher discloses an AttributeFilter 8 that obtains one or more results in response to the query, as follows:

After parsing the document, the "getlength" method is performed to see if the "AttributeFilter" 8 has obtained one or more results in response to the query, and if so, the "getResult" method is performed to obtain one or more results from the "AttributeFilter" 8.
(Kircher, col. 4, lines 24-28).

The Examiner further contends that obtaining the results using Kircher's "getResult" method obtains joined XML fragments (Ans. 15, *see* last two sentences on page). In particular, the Examiner finds that "AttributeFilter 8 (labeled as 'id:AttributeFilter' 8 in Fig. 2) holds the results of the example query" (Ans. 16). Thus, the Examiner concludes that holding all the results of the query is equivalent to joining all the fragments of the document that satisfy the query (Ans. 16, ll. 4-5).

The Examiner proffers that in Kircher's example document, "two ID attributes would be returned satisfying the query." (Ans. 17). The Examiner further contends that the IDs returned are *inherently* multiple attributes (Ans. 17, ll. 14-15).

The Examiner confirms Appellants' assumption in the Appeal Brief (p. 9) that "the examiner is equating the attribute 'id' as being both a fragment (since it is a fragment of the example XML document) and a result

(since the example query returns IDs as shown above in Kircher, col. 3, lines 64-67).” (Ans. 17).

In the Reply Brief, Appellants point out that the Examiner has acknowledged that Kircher’s attribute “id” has been considered by the Examiner as both a fragment and a result (*see* Ans. 17) (Reply Br. 3). Appellants respond that “[e]ven if, *arguendo*, this is an accurate interpretation of Kircher, it clearly fails to disclose forming a set of tuple results, each of which includes multiple fragments, as in claims 1 and 10.” (Reply Br. 3).

ANALYSIS

Independent claims 1, 10, and 19

We consider the Examiner’s rejection of independent claims 1, 10, and 19 as being anticipated by Kircher.

At the outset, the Examiner has not set forth any reasoning in the Answer that it would have been obvious to modify Kircher to achieve a set of tuple results (consisting of joined fragments) that satisfies the query for the stream of mark-up language data (*see* claim 1). Thus, the question of whether Kircher in combination with other references teaches or suggests Appellants’ claimed invention is not before us. Instead, the Examiner has relied upon the Kircher reference as evidence to support an anticipation rejection of each independent claim under 35 U.S.C. § 102(e).

We begin our analysis by noting that Kircher discloses that query filters are first constructed (col. 3, ll. 29-41), and then the filters perform the query (col. 4, l. 6). Kircher discloses that “[t]he filters 2, 6, 8 are, in effect, SAX document handlers.” (col. 4, l. 21).

In particular, we note that Appellants fully disclose in the Specification that the use of SAX (i.e., Simple API for XML) is “known in the art,” as follows:

[0020] XML stream querying system 10 imports the XML data stream into a SAX events API 12, which are *known in the art*, to generate a set of SAX events 17. XML queries 22 are imported into an expression parser 14, which generates a parse tree 15 ("PT") for each query. The parse tree 15 and SAX events 17 are fed into evaluator 16, which uses the SAX events 17 to perform state transitions and populate the buffers 20. Evaluator 16 is also responsible for triggering the tuple construction/buffer management module 18 when the buffers 20 contain enough information to output result tuples 28 [emphasis added].
(Spec. 6, ¶0020).

Returning to the Kircher reference, we note that Kircher discloses “[t]he filters **2, 6, 8** act to ‘filter out’ the event or events that are of interest in response to the query.” (col. 4, ll. 36-37). We further note that Kircher’s filters receive callbacks from parser 4 (col. 4, ll. 7-10). Thus, it appears that the Examiner is reading the claimed “evaluator that receives the parse tree and stream of events, and buffers fragments from the stream of events that meet an evaluation criteria” on at least one of Kircher’s filters (claim 1; *see also* Ans. 4).

Kircher further discloses that the exemplary XML document (col. 3, l. 18-20) is queried “[a]fter the filters **2, 6, 8** have been created and properly registered” (col. 4, ll. 22-23). The “getResult” method is called after the document is parsed (Kircher, col. 4, ll. 24-28).

On the record before us, we find it unclear whether the Examiner has found that the claimed function of joining fragments to form a “set of tuple

results” (as recited in each of independent claims 1, 10, and 19) is performed by Kircher’s ‘getResult’ method that “obtain[s] one or more results from the ‘AttributeFilter’ 8.” (col. 4, l. 28), or whether the Examiner has found that joining of fragments to form a set of tuple results is actually performed by Kircher’s AttributeFilter 8 (col. 4, l. 28). We find the Examiner has not fully developed the record on this point. Moreover, for us to affirm the Examiner on this record would require speculation on our part.

In the latter case (AttributeFilter 8), the Examiner appears to be relying upon a single element (Kircher’s AttributeFilter 8) to perform *both* the claimed evaluation functions and the claimed function of joining of fragments “to form a set of tuple results that satisfies the query for the stream of mark-up language data” (*see* independent claims 1, 10, and 19). In the former case (“getResult’ method), it is our view that merely reading or obtaining one or more results from AttributeFilter 8 (col. 4, ll. 27-28) by calling Kircher’s “getResult’ method does not fairly disclose *joining* fragments *to form a set* of tuple results that satisfies the query for the stream of mark-up language data, as required by the language of independent claims 1, 10, and 19. In particular, we conclude that “reading” or “obtaining” is not the same as “forming” (i.e., constructing).

We are in agreement with Appellants that Kircher discloses a multiple condition query performed by various filters (*see* Kircher, col. 4, ll. 5-21, Fig. 2). After considering the record before us, we find the weight of the evidence supports Appellants’ contention that satisfying a multiple condition query with multiple filters does not fairly disclose bringing two or more items together to form a unit (i.e., a set of tuple results that satisfies the query for the stream of mark-up language data). Moreover, it is our view that

the Examiner has not clearly mapped each element of the claim to the corresponding element found in the Kircher reference, as required to properly support an anticipation rejection under 35 U.S.C. § 102.

Thus, we agree with Appellants that Kircher does not fairly disclose joining fragments “to form a set of tuple results that satisfies the query for the stream of mark-up language data” as claimed (see independent claims 1, 10, and 19). While we agree with the Examiner that the Kircher reference is a close reference that substantially teaches and/or suggests Appellants’ claimed invention, we nevertheless find that Kircher falls short of anticipating Appellants’ claimed invention. “[A]bsence from the reference of any claimed element negates anticipation.” *Kloster Speedsteel AB v. Crucible, Inc.*, 793 F.2d 1565, 1571 (Fed. Cir. 1986).

Because Appellants have shown that the Examiner has failed to meet the burden of establishing a prima facie case of anticipation, we reverse the Examiner’s rejection of independent claim 1 as being anticipated by Kircher. Because each of independent claims 10 and 19 recites equivalent limitations, we also reverse the Examiner’s rejection of these claims as being anticipated by Kircher. Because dependent claim 28 includes all the limitations of independent claim 19 from which it depends, we also reverse the Examiner’s rejection of claim 28 as being anticipated by Kircher. Likewise, we reverse the Examiner’s rejections under 35 U.S.C. § 103 of dependent claims 2-9, 11-18, and 20-27, which depend from independent claims 1, 10, and 19, respectively.

CONCLUSION OF LAW

Based on the findings of facts and analysis above, we conclude that Appellants have shown the Examiner erred in rejecting independent claims 1, 10, 19, and dependent claim 28 under 35 U.S.C. § 102(e) for anticipation. We further conclude that Appellants have shown the Examiner erred in rejecting dependent claims 2-9, 11-18, and 20-27 under 35 U.S.C. § 103(a) for obviousness.

NEW GROUND OF REJECTION

We enter the following new rejection of claims 1-4, 8, 9, 19-22, and 26-28 under the provisions of 37 C.F.R. § 41.50 (b).

35 U.S.C. § 101

Claims 1-4, 8, and 9

Claims 1-4, 8, and 9 are rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter. *See In re Comiskey*, 499 F.3d 1365 (Fed. Cir. 2007). *Comiskey* established that “the application of human intelligence to the solution of practical problems is not in and of itself patentable.” *Id.* at 1379.

Regarding the “system” of independent claim 1, there is no requirement that a computer be used. When we look to Appellants’ disclosure for *context*, we find the Specification broadly discloses that a “system” may be implemented entirely in software:

[0066] It is understood that the systems, functions, mechanisms, methods, and modules described herein can be implemented in hardware, *software*, or a combination of hardware and software [emphasis added].

(Spec. ¶0066).

Thus, we conclude that Appellants' claim 1 broadly encompasses a "system" that (in at least one embodiment) may be implemented entirely in software without integrating a machine, or constituting a process of manufacture, or altering a composition of matter. Regarding the claim term "buffers" in claim 1, we note that this term is used as a verb (i.e., function), and is not used to denote a physical buffer storage medium. Therefore, the nature of the subject matter claimed may be reasonably construed as a mental process since the language of claim 1 broadly encompasses non-tangible embodiments.

We conclude that the language of dependent claims 2-4 does not cure the deficiencies of claim 1. Regarding dependent claims 8 and 9, we construe the claimed "buffer queue" of claim 8 as being directed to providing a data structure, per se. Similarly, we construe the claimed "buffer management *system*" of dependent claim 9 as broadly encompassing at least one embodiment comprised of software alone, as per our discussion above. Because these claims are not tied to a machine or other statutory subject class, we reject claims 1-4, 8, and 9 under 35 U.S.C. § 101 as being directed to non-statutory subject matter.

In contrast, we note that dependent claims 5, 6, and 7 recite "a *work array* for *storing* evaluation data," "a set of output *buffers* to *store* fragments" and, "a set of predicate *buffers* to *store* the content of nodes," respectively. Because claims 5, 6, and 7 recite physical storage and are thus tied to a machine, we consider these claims to be directed to statutory subject matter.

Claims 10-18

We consider claims 10-18 as being directed to statutory subject matter.

Regarding independent claim 10, we note that a “program product stored on a recordable medium” is directed to statutory subject matter so long as the language of claim is not supported in the Specification with non-statutory embodiments (i.e., signals, transmission mediums and the like). *See In re Nuijten*, 500 F.3d 1346, 1357 (Fed. Cir. 2007) (A claim directed to computer instructions embodied in a signal is not statutory under 35 U.S.C. § 101). *Cp. In re Lowry*, 32 F.3d 1579, 1583-84 (Fed. Cir. 1994) (a claim to a data structure stored on a computer readable medium that increases computer efficiency held statutory).

Here, Appellants’ Specification discloses that a “program product” is intended to broadly encompass a program that may be reproduced “in a different material form,” as follows:

[0066] Computer program, software program, program, program product, or software, in the present context mean any expression, in any language, code or notation, of a set of instructions intended to cause a system having an information processing capability to perform a particular function either directly or after either or both of the following: (a) conversion to another language, code or notation; and/or (b) reproduction in a different *material* form. [emphasis added].
(Spec. ¶0066).

Thus, we conclude that a program that may be converted to another language, code or notation and/or be reproduced in “a different material form” (Spec. ¶0066) broadly encompasses reproduction of the (converted)

program in *any* physical tangible form that would *exclude* signals and other non tangible transmission mediums. Therefore, we conclude that claims 10-18 are directed to statutory subject matter.

Claims 19-22 and 26-28

Claims 19-22 and 26-28 are rejected under 35 U.S.C. § 101 as being directed to non-statutory subject matter.

We note that independent claim 19 is directed to a “method of querying a stream of mark-up language data” It is our view that the method of claims 19-22 and 26-28 suffers from the same deficiencies as discussed above regarding claims 1-4, 8, and 9.

In contrast, we note that dependent claims 23-25 recite an *array* or *buffers* that *store* information. Because claims 23-25 recite physical storage and are thus tied to a machine, we consider these claims to be directed to statutory subject matter.

DECISION

The decision of the Examiner rejecting claims 1-28 is reversed.

This decision contains a new ground of rejection pursuant to 37 C.F.R. § 41.50(b) (effective September 13, 2004, 69 Fed. Reg. 49960 (August 12, 2004), 1286 Off. Gaz. Pat. Office 21 (September 7, 2004)). 37 C.F.R. § 41.50(b) provides “[a] new ground of rejection pursuant to this paragraph shall not be considered final for judicial review.” 37 C.F.R. § 41.50(b) also provides that the Appellants, WITHIN TWO MONTHS FROM THE DATE OF THE DECISION, must exercise one of the

following two options with respect to the new ground of rejection to avoid termination of the appeal as to the rejected claims:

(1) *Reopen prosecution*. Submit an appropriate amendment of the claims so rejected or new evidence relating to the claims so rejected, or both, and have the matter reconsidered by the examiner, in which event the proceeding will be remanded to the examiner. . . .

(2) *Request rehearing*. Request that the proceeding be reheard under § 41.52 by the Board upon the same record. . . .

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a)(1)(iv).

REVERSED - 37 C.F.R. § 41.50(b)

rwk

HOFFMAN WARNICK & D'ALESSANDRO, LLC
75 STATE STREET
14TH FLOOR
ALBANY NY 12207