NOTE: This disposition is nonprecedential.

United States Court of Appeals for the Federal Circuit

THOUGHT, INC., A CALIFORNIA CORPORATION,
Plaintiff-Appellant

v.

ORACLE CORPORATION, A DELAWARE CORPORATION, ORACLE AMERICA, INC., A DELAWARE CORPORATION, ORACLE INTERNATIONAL CORPORATION, A CALIFORNIA CORPORATION,
Defendants-Appellees

2016-2369

Appeal from the United States District Court for the Northern District of California in No. 3:12-cv-05601-WHO, Judge William H. Orrick, III.

Decided: August 21, 2017

MARK CARLSON, Hagens, Berman, Sobol, Shapiro LLP, Seattle, WA, argued for plaintiff-appellant. Also represented by STEVE BERMAN; JEFF D. FRIEDMAN, Berkeley, CA; KELLY G. HYNDMAN, CHANDRAN B. IYER, Sughrue Mion, PLLC, Washington, DC.

STEVEN MOORE, Kilpatrick Townsend & Stockton LLP, San Francisco, CA, argued for defendants-appellees. Also represented by GIA L. CINCONE, GREGORY PHILIP FARNHAM, BENJAMIN M. KLEINMAN-GREEN.

Before MOORE, SCHALL, and O'MALLEY, *Circuit Judges.*

MOORE, *Circuit Judge*.

Thought, Inc. ("Thought") appeals from the Northern District of California's summary judgment of noninfringement of claims 1, 3, 5, 7, and 8 of U.S. Patent No. 5,857,197 ("the '197 patent"). For the reasons discussed below, we *affirm*.

BACKGROUND

The '197 patent, titled "System and Method for Accessing Data Stores as Objects," discloses a system and method for object-oriented programs to access data in a relational database. The type of system disclosed is commonly known as "middleware." Prior art techniques utilized customized code for each relational table, which was costly and time-consuming to create and maintain. The '197 patent utilizes an abstraction layer with a set of interchangeable runtime adapters using the same application programming interface, "effecting a consistent interface to the data store regardless of its underlying structure." '197 patent at abstract. Claim 3 is representative:

> A system for accessing at least one data store having a data store content and a data store schema as at least one object from at least one object application comprising:
>
> > at least one object schema including meta data corresponding to the data store schema;

> a first adapter responsive to the object application including an application bridge receiving an object comprising object attributes and an object name from the object application, said first adapter *extracting the object attributes and the object name from the object* to effect packing of the object attributes and the object name as data, said first adapter unpacking the data to effect instantiating the object attributes and the object name into a new object; and
>
> a second adapter in communication with said first adapter and in communication with at least one data store, said second adapter having a meta data map comprising at least one object name and providing the data store content from at least one data store corresponding to the object attributes and the meta data.

Id. at 35:26–39 (emphasis added).

Thought sued Oracle Corp. ("Oracle") in the Northern District of California, alleging that Oracle's TopLink program and related applications infringe claims 1, 3, 5, 7, and 8 of the '197 patent. Oracle denied infringement and asserted a counterclaim of invalidity. The district court issued a claim construction order in which it construed "object" as an "instance of a class." J.A. 116. It did not construe the term "extracting."

Oracle moved for summary judgement of noninfringement and invalidity. The district court granted summary judgment of noninfringement. It held, *inter alia*, that Thought failed to raise a material issue of fact that the accused software performed the claim limitation "extracting the object attributes and the object name from the object" for two independent reasons. First, it held

that the claim language and specification make clear that the term "extracting" refers to "extracting a *subset* of information from an object." *Thought, Inc. v. Oracle Corp.*, No. 12-CV-05601-WHO, 2016 WL 3230696, at *13 (N.D. Cal. June 13, 2016) (emphasis in original). Second, in light of the construction of "object" as an instance of a class, it held that Thought did not demonstrate that the accused software extracts the object name and object attributes "from a singular 'instance of a class.'" *Id.* The district court dismissed Oracle's counterclaim without prejudice. Thought timely appealed. We have jurisdiction pursuant to 28 U.S.C. § 1295(a)(1).[1]

DISCUSSION

We review the district court's ultimate claim construction de novo, and we review any subsidiary fact findings for clear error. *Teva Pharm. USA, Inc. v. Sandoz, Inc.*, 135 S. Ct. 831, 841–42 (2015). Although we apply our own law with respect to issues of substantive patent law, we review the grant or denial of summary judgment using the law of the relevant regional circuit. *Accenture Glob. Servs., GmbH v. Guidewire Software, Inc.*, 728 F.3d 1336, 1340–41 (Fed. Cir. 2013). The Ninth Circuit reviews the district court's grant or denial of summary judgment de novo. *JL Beverage Co., LLC v. Jim Beam Brands Co.*, 828 F.3d 1098, 1104 (9th Cir. 2016).

Thought's only infringement theory on appeal is that the TopLink program's `find()` method meets the "extracting the object attributes and the object name from the object" claim limitation. The `find()` method receives two separate parameters: `Class<T> entityClass` and

[1] Oracle did not cross-appeal or otherwise object to the district court order dismissing its invalidity counterclaim. We therefore do not review the propriety of the district court's sua sponte dismissal of that counterclaim.

`Object primaryKey`. The `find()` method does not pull information from the `primaryKey` object. Rather, it merely passes along `primaryKey` to another method wholesale. J.A. 876–78. Thought alleges that the "'object name' in the form of Class<T> entityClass is extracted from the object that is the set of parameters passed into the find() method" and the "'object attributes' in the form of Object primaryKey are extracted from the object that is the set of parameters passed into the find() method." J.A. 876–77.

Oracle argues the `find()` method cannot meet the "extracting" limitation because wholesale copying or passing along of a reference to an object is not part of the plain and ordinary meaning of "extracting the [data] from the object." It argues the claims require the first adapter to extract something less than the entire object. Thought argues that nothing in the patent requires limiting "extracting" to a subset of information and that a person of ordinary skill in data processing would have understood "extracting" to mean "obtaining."

We agree with the district court and Oracle that the plain and ordinary meaning of "extracting . . . from the object" cannot mean merely passing along or copying the entire object, including the container of the thing extracted. The full "extracting" clause of claim 3 claims "said first adapter extracting the object attributes and the object name from the object to effect packing of the object attributes and the object name as data." '197 patent at 35:33–36. The plain language and context of this clause demonstrates that the word "from" indicates the source from which the extracted thing is taken. *See Phillips v. AWH Corp.*, 415 F.3d 1303, 1314 (Fed. Cir. 2005) (en banc), *cert. denied*, 546 U.S. 1170 (2006) ("[T]he context in which a term is used in the asserted claim can be highly instructive."). Just as a child might "extract" all of the cookies from the cookie jar and leave the cookie jar itself behind, the first adapter may extract all of the data

contained within the object but must leave behind the data container of the object itself. Taking, or making an exact copy of, the container and all of the contents held within the container is not extracting the contents from the container.

The claim, read in the context of the specification, strongly supports this conclusion because it indicates that the extraction is performed in order to reorganize the object attributes and object name as "data" separate from the object. The "extracting" limitation closely tracks the specification language used to describe the only disclosed embodiment that "extracts" in any way: "The first adapter 400 then extracts the object attributes 103 and the object name 104 from the object 102, and packs the object attributes 103 and the object name 104 as data 105 to be used in communication and transport layers." '197 patent at 7:42–46. The claim language and the associated portion of the specification indicate that the purpose of the extraction is to pack the data extracted from the object as data for improved communication. The specification also explains that the disclosed "method of breaking down objects 102 (112) into the corresponding primitive types comprising data 105 (115) ensures successful transfers of any kind of object irrespective of object application 101 views of the object(s)' data elements." *Id.* at 4:61–65. The disclosed purpose of the extraction limitation, therefore, is to separate the object attributes and object name from the object itself to effect the '197 patent's aim of "a simple and consistent interface to at least one data store(s) regardless of its underlying structure." *Id.* at 3:31–35.

The use of the terms "extracting" and "obtaining" in other claims further supports the conclusion that "extracting" does not include the container of the data. *See Phillips*, 415 F.3d at 1314 ("[T]he usage of a term in one claim can often illuminate the meaning of the same term in other claims."). Like claim 3, claim 7 claims "extracting the object attributes and the object name from the object,"

but in another limitation it also claims "*obtaining* data store content and/or an execution status." '197 patent at 36:15–16 (emphasis added). In the specification's description of this claimed embodiment, the second adapter executes a command with the accessed database "and obtains the data store content 301 and an execution status 306 based on executing at least one such command 303." *Id.* at 8:1–5. The second adapter "then processes the data store content 304 and the execution status 306 using meta data 201, and packs the obtained data store content 304 and the execution status 306 as data 115." *Id.* at 8:6–9. Unlike the first adapter's extraction of the object attributes and object name, which are immediately packed as data, the claimed second adapter *processes* the data store content and/or execution status after they are obtained and before they are packed. *Id.* This context indicates that unlike extracted data, obtained data might be encapsulated in a container that must be processed before being packed into data suitable for communication to the first adapter. *See id.* at 8:9–10 ("The second adapter 500 communicates the data 115 to the first adapter 400.").

Oracle also argues that because the `find()` method allegedly draws the object name from one object (`entityClass`) and the object attributes from another object (`primaryKey`), the alleged software cannot extract information from a single object as required by the claims. Thought argues that the set of parameters to the `find()` method is itself an object that contains references to two other objects from which the object name and object attributes are extracted. Thought does not dispute that the claim language requires the object name and object attributes be extracted from a single object, or instance of a class. Instead, it argues that there is a single "set of parameters object" passed to the `find()` method that contains references to two other objects—one for the

object name, the other for the object attributes. Thought Reply Br. 14–18, *see also* J.A. 345–46, 774, 876–77.

We agree with the district court and Oracle that the accused method cannot extract two pieces of data "from the object" when each piece of data is derived from a separate object. Thought and its expert admit that under its theory of infringement, "the 'object' from the object application is an object with references to two other objects, not the two objects themselves." J.A. 346 ¶ 62. The claim language requires that object attributes and object name be extracted "*from the object*." The antecedent basis for "the object" indicates that the object "compris[es] object attributes and an object name." '197 patent at 35:32–33. A set of parameters containing nothing more than references to two separate objects, from which the object name and object attributes are separately extracted, cannot meet the express claim language. The singular "object" required by the claim cannot be comprised of the object attributes and object name if it is merely a set of references to separate objects that comprise the object attributes and object name.

The district court correctly granted summary judgment of noninfringement based on both rationales regarding the "extracting" limitation. We do not reach the other alternative rationales on which the district court granted summary judgment. *See Thought*, 2016 WL 3230696, at *9–12.

CONCLUSION

For the foregoing reasons, we *affirm* the district court's summary judgment of noninfringement of claims 1, 3, 5, 7, and 8 of the '197 patent.

AFFIRMED