

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

SERVICENOW, INCORPORATED,
Petitioner,

v.

INQUISIENT INCORPORATED,
Patent Owner.

IPR2023-01071
Patent 7,979,468 B2

Before BRIAN J. MCNAMARA, GREGG I. ANDERSON, and
CHARLES J. BOUDREAU, Administrative Patent Judges.

BOUDREAU, *Administrative Patent Judge.*

DECISION
Denying Institution of *Inter Partes* Review
35 U.S.C. § 314

I. INTRODUCTION

ServiceNow, Inc. (“Petitioner”) filed a Corrected Petition (Paper 3, “Pet.”) requesting *inter partes* review of claims 1–39 of U.S. Patent No. 7,979,468 B2 (Ex. 1001, “the ’468 patent”). InQuisient Inc. (“Patent Owner”) filed a Preliminary Response (Paper 7, “Prelim. Resp.”). With the Board’s authorization, Petitioner filed a Preliminary Reply to the Preliminary Response (Paper 8, “Pet. Prelim. Reply”), and Patent Owner filed a Sur-reply to the Preliminary Reply (Paper 9, “PO Prelim. Sur-reply”).

We have jurisdiction under 35 U.S.C. § 6 (2018). Under 35 U.S.C. § 314(a), an *inter partes* review may not be instituted unless the information presented in the Petition and any response thereto shows “there is a reasonable likelihood that the petitioner would prevail with respect to at least 1 of the claims challenged in the petition.” Considering the Petition, Preliminary Response, Preliminary Reply, Sur-reply, and all supporting evidence, we determine that Petitioner does not demonstrate a reasonable likelihood of prevailing for any of the challenged claims. Accordingly, we do not institute *inter partes* review.

A. *Real Parties in Interest*

Petitioner and Patent Owner identify themselves as the real parties in interest in this proceeding. Pet. 69; Paper 6, 1 (Patent Owner’s Mandatory Notices).

B. *Related Matter*

The parties identify *InQuisient Inc. v. ServiceNow, Inc.*, 1:22-cv-00900 (D. Del.) (“Delaware Lawsuit”) as involving the ’468 patent. Pet. 1; Paper 6, 1.

C. The '468 patent

The '468 patent “relates to methods and systems for storing, managing, indexing, interrelating, and/or retrieving data sets.” Ex. 1001, 1:13–16. The '468 patent states that a conventional system implements its own data model and rules according to a user interface, requiring developers to write dedicated code and user interface code that displays or manages data. *Id.* at 1:20–26. The '468 patent seeks to provide a system and method to store, manage, index, interrelate, and/or retrieve data sets in a manner that is independent of the data model. *Id.* at 1:50–52. The system includes a processor and data repository. *Id.* at 1:59–63. An exemplary entity-relationship diagram of a elements metadata repository (EMR) shown in Figure 1 includes eight modules recited in claim 1. *Id.* at 3:1–4. A database creator enters database information in the modules rather than writing dedicated code and user interface code. *Id.* at 8:50–60. In this elements-based approach, each of the elements conforms to rules of generalized things in the universe, e.g., through entries in an element relation table. *Id.* at 9:18–23.

The challenged claims recite a data repository that includes an element module configured to store and uniquely identify a plurality of elements each identified by a unique element identification. Ex. 1001, 1:66–2:1, 6:9–29 (describing elements as uniquely identified by an ID, name, and class identifier that points to an indicator of the type or classification of the element). The EMR places “things” into an element table—an element stored in an element table implies that the user can find similarity between any “thing” the user would like to discuss, report, store, or manage, without regard to their exact nature. *Id.* at 5:66–6:6.

The data repository also includes an element relation module configured to store one or more relationships between the elements in the element module. Ex. 1001, 2:1–3, 6:30–40 (describing an element relation table that stores relationships between elements, e.g., each entry in the element relation table contains a pointer to at least two elements in the elements table and a type definition describing the type of relationship).

A class module is configured to define at least one class of the elements and store the class. Ex. 1001, 1:64–2:5. The EMR also includes an attribute module configured to define and store the attributes and a class attribute module configured to define and store one or more class-attribute associations between at least one of the attributes and the class, and a type definition module configured to define and store one or more types of the class, the attributes related to the class, and the relationships between the elements. *Id.* at 2:4–9. An attribute table 110 holds the definition of each attribute, e.g., the physical name and type of data that corresponds to the database column name and column type where the actual values are stored. *Id.* at 5:1–5. A class attribute table may associate a given attribute with one or more classes in a class table. *Id.* at 5:5–6.

A type definition module is configured to define and store one or more types of the class, the attributes related to the class and the relationship between the elements. Ex. 1001, 2:9–12.

The EMR can utilize entries in a named_text table to name structured query language (SQL) queries and filter query entries into one or more dynamic folders, such that the result of a query may vary over time. Ex. 1001, 10:59–11:34.

D. Illustrative Claim

Of the challenged claims, claims 1, 14, and 27 are independent. Claim 1 is reproduced below with paragraph designations used in the Petition.

1[pre]. A computerized system for manipulating data sets, comprising:

[a] a processor; and

[b] a data repository, the data repository being adapted to process, retrieve, and store data contained in the data sets and one or more layers of metadata of the data in the data sets, comprising:

[c] an element module configured to store and uniquely identify a plurality of elements, wherein each of the elements is identified by a unique element identification;

[d] an element relation module configured to store one or more relationships between the elements in the element module;

[e] a class module configured to define at least one class of the elements and store the class;

[f] an attribute module configured to define one or more attributes and store the attributes;

[g] a class attribute module configured to define and store one or more class-attribute associations between at least one of the attributes and the class;

[h] a type definition module configured to define and store one or more types of the class, the attributes related to the class, and the relationships between the elements;

[i] a state machine module configured to store one or more state machine types associated with at least one of the elements; and

[j] a status module configured to store one or more statuses of each state machine.

Ex. 1001, 16:13–41; Pet. vi. Independent claim 14 recites a method with limitations generally corresponding to those of claim 1. Ex. 1001, 17:30–18:53. Independent claim 27 recites a non-transitory computer readable medium with limitations generally corresponding to those of claim 1. *Id.* at 18:33–58.

E. Asserted Grounds of Unpatentability

Petitioner asserts the following grounds of unpatentability:

Claim(s) Challenged	35 U.S.C. §¹	Reference(s)/Basis
1–9, 11–22, 24–35, 37–39	103(a)	Althoff, ² Nye ³
10, 23, 36	103(a)	Althoff, Nye, Surjanto ⁴
6, 7, 19, 20, 32, 33	103(a)	Althoff, Nye, Moore ⁵

Pet. 1. In addition to the references listed above, Petitioner relies on a Corrected Declaration of Dr. Jules White (Ex. 1003).

II. ANALYSIS

A. Level of Ordinary Skill in the Art

In determining the level of skill in the art, we consider the type of problems encountered in the art, the prior art solutions to those problems, the rapidity with which innovations are made, the sophistication of the

¹ The Leahy-Smith America Invents Act (“AIA”) included revisions to 35 U.S.C. § 103 that became effective on March 16, 2013. Because the ’468 patent issued from an application filed before March 16, 2013, we apply the pre-AIA version of § 103.

² Althoff et al., U.S. Patent No. 6,374,252 B1, issued April 16, 2002 (Ex. 1006).

³ Nye, U.S. Patent No. 6,341,279 B1, issued January 22, 2002 (Ex. 1005).

⁴ Surjanto, B. et al., *XML Content Management Based on Object-Relational Database Technology*, Proc. First Int’l Conf. on Web Info. Sys. Eng’g 70 (IEEE Comput. Soc’y 2000) (Ex. 1008).

⁵ Moore et al., U.S. Patent No. 7,627,552 B2, issued December 1, 2009 (Ex. 1007).

technology, and the educational level of active workers in the field. *Custom Accessories, Inc. v. Jeffrey-Allan Indus. Inc.*, 807 F.2d 955, 962 (Fed. Cir. 1986); *Orthopedic Equip. Co. v. U.S.*, 702 F.2d 1005, 1011 (Fed. Cir. 1983).

Petitioner contends that a person of ordinary skill in the art at the time of the invention of the '468 patent “would have [had] a Bachelor’s degree in Computer Science and three to five years of industry experience in application development, database development, enterprise application management; or a Ph.D. specializing in the field,” but that “more education could compensate for less experience and vice versa.” Pet. 3 (citing Ex. 1003 ¶ 24).

Patent Owner does not contest Petitioner’s contention with regard to the level of skill. *See generally* Prelim. Resp.

As Petitioner’s unchallenged description of the level of ordinary skill appears appropriate to the subject matter of the '468 patent and the cited prior art, we apply it for purposes of this Decision. *See Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001) (the prior art may reflect an appropriate level of skill in the art).

B. Claim Construction

We interpret claim terms using “the same claim construction standard that would be used to construe the claim in a civil action under 35 U.S.C. 282(b).” 37 C.F.R. § 42.100(b) (2023). In this context, claim terms “are generally given their ordinary and customary meaning” as understood by a person of ordinary skill in the art in question at the time of the invention. *Phillips v. AWH Corp.*, 415 F.3d 1303, 1312–13 (Fed. Cir. 2005) (citations omitted) (en banc). “In determining the meaning of the disputed claim limitation, we look principally to the intrinsic evidence of record, examining the claim language itself, the written description, and the prosecution

history, if in evidence.” *DePuy Spine, Inc. v. Medtronic Sofamor Danek, Inc.*, 469 F.3d 1005, 1014 (Fed. Cir. 2006) (citing *Phillips*, 415 F.3d at 1312–17). Extrinsic evidence is “less significant than the intrinsic record in determining ‘the legally operative meaning of claim language.’” *Phillips*, 415 F.3d at 1317 (citations omitted).

Any special definition for a claim term must be set forth in the specification with reasonable clarity, deliberateness, and precision. *In re Paulsen*, 30 F.3d 1475, 1480 (Fed. Cir. 1994).

We construe only those claim terms that require analysis to determine whether to institute *inter partes* review. *See Realtime Data, LLC v. Iancu*, 912 F.3d 1368, 1375 (Fed. Cir. 2019) (“The Board is required to construe ‘only those terms . . . that are in controversy, and only to the extent necessary to resolve the controversy’” (quoting *Vivid Techs., Inc. v. Am. Sci. & Eng’g, Inc.*, 200 F.3d 795, 803 (Fed. Cir. 1999))).

Noting that the ’468 patent specification describes a data repository as including modules, Petitioner contends that a person of ordinary skill would have understood a “‘module’ to be a portion of a data repository.” Pet. 4. Petitioner notes that Patent Owner conceded in the Delaware Lawsuit that “an example of a ‘module’ is ‘a table or a portion of a table’ (i.e., a data repository).” *Id.* (citing Ex. 1013 (Delaware Lawsuit, Transcript of Motions to Dismiss), 78). Petitioner then states that it “applies Patent Owner’s own definition of a ‘module’ throughout this Petition.” *Id.*

Patent Owner argues that, in the context of the ’468 patent, the term “module” would be understood to be a table-based structure, i.e., “a module comprises one or more columns within one or more tables.” Prelim. Resp. 19. According to Patent Owner, to the extent the panel issues a formal construction, “module” should be construed to mean “a table or, one or more

columns of/within one or more tables.” *Id.* at 20. Patent Owner does not cite to any definition of “module” in the ’468 patent specification or any disclosure that would limit a “module” to a table or a column of a table.

Patent Owner further argues that we should deny institution and “not reach the substance of the dispute” because the Petition does not apply what Petitioner contends to be the proper construction of the term “module.”

Prelim. Resp. 21–25. According to Patent Owner, because Petitioner does not apply what it believes is the correct construction, but instead applies a construction Petitioner alleges Patent Owner proposed in the Delaware Lawsuit, Petitioner has failed to comply with the requirements of 37 C.F.R. § 42.104(b)(3). *Id.* at 22 (citing *Toyota Motor Corp. v. Blitzsafe Tex., LLC*, IPR2016-00422, Paper 12, 26 (PTAB Jul. 6, 2016)).

In its Preliminary Reply, Petitioner notes that in response to a question during a hearing on a Motion to Dismiss under 35 U.S.C. § 101 in the Delaware Lawsuit, Patent Owner argued the claim terms recite structure because the claimed module is a table or a portion of a table. Pet. Prelim. Reply 1–2 (citing Ex. 1013, 78). Petitioner also notes that in a Joint Claim Construction Chart submitted to the District Court, Patent Owner proposed “‘module’ be given its ‘Plain and Ordinary Meaning,’” without defining such plain and ordinary meaning, and that Petitioner proposed that “module” be construed to mean “a table or, one or more columns of/within a single table.” *Id.* at 2 (citing Ex. 1011 (Delaware Lawsuit, Joint Claim Construction Chart), 2).

The premise of Patent Owner’s argument is flawed. Petitioner asserts that, in the context of the ’468 patent, a person of ordinary skill would understand “module” to be a portion of a data repository. Pet. 4. Petitioner also states its understanding that Patent Owner described a table or a portion

of a table in the Delaware Lawsuit as an *example* of a module, i.e., an example of a portion of a data repository. *Id.*; *see* Ex. 1013, 78:13–19 (“This claim has a very specific data dictionary. It’s got particular structures. The modules that are referred to in the claim, an *example* that’s used in the specification is a table or a portion of a table. A table is a structure that people know how to build.”) (emphasis added); *see also* PO Prelim. Sur-reply 1–2 (Patent Owner stating that at the hearing on the Motion to Dismiss in the Delaware Lawsuit, Patent Owner referred to “an example that’s used in the specification is a table or portion of a table”).

Petitioner’s acknowledgement of Patent Owner’s argument in the Delaware Lawsuit does not change Petitioner’s recognition that the ’468 patent describes a data repository as including modules, such that an ordinarily skilled person would have understood a “module” to be a portion of a data repository. The ’468 patent includes an extensive discussion of tables and relationships between tables, but does not define the term “module.” In the ’468 patent, the term “module” appears in the claims; “module” also appears in the “SUMMARY,” in a manner that parrots the language of the claims. Ex. 1001, 1:65–2:12.

The term “module” is a classic nonce word that does not connote definite structure. *See, e.g., Williamson v. Citrix Online, LLC*, 792 F.3d 1339, 1350 (Fed. Cir. 2015) (“‘Module’ is a well-known nonce word that can operate as a substitute for ‘means’ in the context of [pre-AIA 35 U.S.C.] § 112, para. 6” and “is simply a generic description for software or hardware that performs a specified function” that “does not provide any indication of structure.”). Independent claims 14 and 27 do not use the term “module,”

instead reciting limitations similar to those of claims 1–13 using method steps. Ex. 1001, 17:30–53, 18:33–58.⁶

Petitioner merely recognizes that Patent Owner attempted to avoid a finding in the Delaware Lawsuit that claims 1–39 recite unpatentable subject matter or are indefinite by ascribing structure to the term “module.” Patent Owner does not, indeed cannot, contend that the structure it identifies as a module, i.e., a table or a portion of a table, is not an example of a portion of a data repository. Petitioner’s efforts to address the structure Patent Owner ascribes to the term “module” are not inconsistent with Petitioner’s proposed construction.

To the extent that claims 1–39 can be construed as definite, we construe “module” to mean a portion of a data repository, and note that an example of a structure of a module disclosed in the ’468 patent, i.e., a portion of a data repository, is a table or portion of a table.

C. *Principles of Law*

“In an [*inter partes* review], the petitioner has the burden from the onset to show with particularity why the patent it challenges is unpatentable.” *Harmonic Inc. v. Avid Tech., Inc.*, 815 F.3d 1356, 1363 (Fed. Cir. 2016) (citing 35 U.S.C. § 312(a)(3) (requiring *inter partes* review petitions to identify “with particularity . . . the evidence that supports the

⁶ The method steps of claims 14 and 27 may also be construed under 35 U.S.C. § 112 paragraph 6 as means plus function limitations. *See O.I. Corp. v. Tekmar Co.*, 115 F.3d 1576, 1583 (Fed. Cir. 1997) (recognizing that although “[t]he mere fact that a method claim is drafted with language parallel to an apparatus claim with means-plus-function language does not mean that the method claim should be subject to an analysis under § 112, paragraph 6[,] . . . each limitation of each claim must be independently reviewed to determine if it is subject to the requirements of § 112, paragraph 6”).

grounds for the challenge to each claim’’)). This burden of persuasion never shifts to Patent Owner. *See Dynamic Drinkware, LLC v. Nat’l Graphics, Inc.*, 800 F.3d 1375, 1378 (Fed. Cir. 2015) (discussing the burden of proof in *inter partes* review).

The question of obviousness is resolved on the basis of underlying factual determinations including (1) the scope and content of the prior art; (2) any differences between the claimed subject matter and the prior art; (3) the level of ordinary skill in the art; and (4) objective evidence of obviousness or nonobviousness. *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966). Additionally, the obviousness inquiry typically requires an analysis of “whether there was an apparent reason to combine the known elements in the fashion claimed by the patent at issue.” *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 418 (2007) (citing *In re Kahn*, 441 F.3d 977, 988 (Fed. Cir. 2006) (requiring “articulated reasoning with some rational underpinning to support the legal conclusion of obviousness’’)); *see In re Warsaw Orthopedic, Inc.*, 832 F.3d 1327, 1333 (Fed. Cir. 2016) (citing *DyStar Textilfarben GmbH & Co. Deutschland KG v. C.H. Patrick Co.*, 464 F.3d 1356, 1360 (Fed. Cir. 2006)).

An obviousness analysis “need not seek out precise teachings directed to the specific subject matter of the challenged claim, for a court can take account of the inferences and creative steps that a person of ordinary skill in the art would employ.” *KSR*, 550 U.S. at 418; *accord In re Translogic Tech., Inc.*, 504 F.3d 1249, 1259 (Fed. Cir. 2007). Petitioner cannot satisfy its burden of proving obviousness by employing “mere conclusory statements.” *In re Magnum Oil Tools Int’l, Ltd.*, 829 F.3d 1364, 1380 (Fed. Cir. 2016). Instead, Petitioner must articulate a reason why a person of

ordinary skill in the art would have combined the prior art references. *In re NuVasive*, 842 F.3d 1376, 1382 (Fed. Cir. 2016).

A reason to combine or modify the prior art may be found explicitly or implicitly in market forces; design incentives; the “interrelated teachings of multiple patents”; “any need or problem known in the field of endeavor at the time of invention and addressed by the patent”; and the background knowledge, creativity, and common sense of the person of ordinary skill. *Perfect Web Techs., Inc. v. InfoUSA, Inc.*, 587 F.3d 1324, 1328–29 (Fed. Cir. 2009) (quoting *KSR*, 550 U.S. at 418–21).

In determining whether a claim is obvious in light of the prior art, we consider any relevant objective evidence, when in evidence, of obviousness or non-obviousness. *See Graham*, 383 U.S. at 17. At this stage of the proceeding, the parties do not present evidence of such objective considerations.

We analyze the asserted grounds of unpatentability in accordance with these principles to determine whether Petitioner has met its burden to establish a reasonable likelihood of success at trial.

D. Obviousness of Claims 1–9, 11–22, 24–35, and 37–39 over Althoff and Nye

Petitioner contends that claims 1–9, 11–22, 24–35, and 37–39 are unpatentable over the combined teachings of Althoff and Nye. Pet. 10–51. Patent Owner disputes Petitioner’s contentions. Prelim. Resp. 25–56.

1. Althoff

Althoff relates to modeling of object-oriented database structures, translation to relational database structures, and dynamic searches of the same. Ex. 1006, code (57), 1:16–18, 3:12–14. By way of background, Althoff discloses that it is known in software systems to associate data items

and data types to allow elements of a system to rely on the characteristics of the data type and relationships between one data type and other data types. Ex. 1006, 1:20–28. Althoff further discloses that users can define a class, e.g., telephone number, to allow elements of the system to store, manipulate, or retrieve telephone numbers as though they were fundamental pieces of information. *Id.* at 1:30–35. Althoff explains that it has been found that the technique of defining classes of software objects and manipulating those objects also has been useful for database applications. *Id.* at 1:41–43. In particular, according to Althoff, in object-oriented database applications, a user can define classes of software objects, as well as properties of those classes and relationships between them, and can populate a database with data items that are instances of those objects. *Id.* at 1:43–47. For example, Althoff discloses, the class “telephone number” might be used to store telephone numbers for person, companies, and computer networks. *Id.* at 1:47–50. Althoff explains that it has been found that object-oriented database management provides advantages such as rapid application and database development and relative software reliability. *Id.* at 1:50–53. Notwithstanding such advantages, however, Althoff further discloses that a problem has arisen in the art insofar as “many database systems have been designed to operate with another, different database technique, known as ‘relational database’ management.” *Id.* at 1:54–57.

In a relational database, Althoff explains,

the database comprises a set of data tables which represent the relationships between data items. Each table comprises a set of records, one record for each data item; each record comprises a set of columns or fields, one column for each data value associated with the data item. (For example, each record in a table of persons might have a column defined for that person’s telephone number; similarly, each record in a table of companies

might have a column defined for that company's telephone number.)

Ex. 1006, 1:58–66. According to Althoff, relational databases have been found to be successful and efficient in managing large databases, and accordingly, there has been substantial investment in relational database management systems and the applications and computer programs that support them. *Id.* at 1:66–2:4. Therefore, Althoff discloses, it would be advantageous to obtain the advantages of object-oriented database management while retaining the efficiency of and installed investment in relational database management systems, but tools for creating, implementing, and manipulating object-oriented databases are generally inconsistent with the tools for creating, implementing, and manipulating relational databases. *Id.* at 2:5–8. Althoff thus seeks to provide a method of compiling and translating between object-oriented and relational database structures that would obtain the advantages of the former while retaining the ability to use the then-existing relational database management systems. *Id.* at 3:4–9.

To achieve the stated goal, Althoff provides a method and system that allow a user to create, edit, and manipulate an object database, objects for that object database, and queries to be applied to the object database, where the method and system dynamically translates the object database, the objects for the object database, and queries to be applied to the object database into a set of relational database structures, data for those relational database structures, and queries to be applied to those relational database structures, respectively. Ex. 1006, 3:12–25. According to Althoff, the method and system “include[] a meta-model of the user's object database, which is itself an object database, and which has itself been translated into a

set of relational database structures for manipulation by a relational database engine.” *Id.* at 3:26–30. The meta-model comprises a set of classes, objects, and relationships between classes that model the classes and relationships between classes of the system. *Id.* at 3:30–32. “Each of these classes may comprise a set of searchable properties, and each of these relationships may comprise an inheritance relationship (between a base class and a derived class) or a data-model relationship (such as a one-to-one, one-to-many, or many-to-many relationship).” *Id.* at 3:31–37. Further, “[t]he data model of the user’s object database is modeled by actual objects in the meta-model, and editing or manipulating the user’s object database is modeled by creating, modifying, or deleting objects in the meta-model.” *Id.* at 3:37–41. “The meta-model also models itself, in the same manner as it models the user’s object database, and may be manipulated in the same manner as the user’s object database.” *Id.* at 3:41–44.

In a preferred embodiment, each class in the user’s object database is modeled by a relational database table, each searchable property of that class is modeled by a column in that table, and each object of that class is modeled by a row in that table and corresponding rows in tables representing base classes for that object. Each such table comprises an “object-ID” column generated by the system, comprising a unique ID for each object in the system. Each relationship between two objects is modeled by providing a pointer from the first object to the second object; the pointer comprises a column (generated by the system to model the relationship) in the first object’s table with the object-ID of the second object. These relational database structures are created, modified, or deleted dynamically or incrementally in response to user commands to create, edit, and manipulate the user’s object database.

Id. at 3:44–60. Thus, each class may comprise a set of searchable properties and the relationship may be an inheritance relationship (between a base class

and a derived class) or a data-model (e.g., a one-to-one, many-to-one, or many-to-many relationship). *Id.*; *see also id.* at 7:21–25 (describing the meta-model as “comprising an object database having as its objects the classes 101 in the user’s object database 100, searchable properties 102 of those classes, and relationships 103 between those classes”).

Metamodel 200 comprises classes shown in Table 3-1. Ex. 1006, 10:41–11:10. Each class of the user’s object database 100 is modeled by an object of class Class 301. Ex. 1006, 11:13–14. Each object of class Class 301 includes an object ID that is unique for the object, a class label (a name to be displayed for the class), a class DB name (to be used by the system for generating SQL commands and accessing relational structures corresponding to this class 101), properties 102, such as Class Group, Class Type, and Is System Specific that maintain information about the class 101 in system 200, as well as information about other properties. *Id.* at 11:12–12:23.

2. Nye

Nye, titled “Method and Apparatus for Event Modeling,” “provides a method that allows a developer to add complex dependency logic to an existing database without having to modify the underlying structure of the database.” Ex. 1005, codes (54), (57), 5:34–37. In one embodiment, an “event model” is used to provide a way to flexibly handle record state transitions. *Id.* at code (57), 5:41–42. Nye explains that an event model is “a set of one or more items called an event,” where “[a]n event can represent anything it is defined to represent.” *Id.* at code (57), 5:43–44, 5:54–55, 10:58–61. According to Nye, each event has an associated event type and contains dependency logic that interrelates events with one another. *Id.* at code (57), 5:44–46. Further, “[e]ach event represents a set of actions that

are optionally contingent upon a condition,” where “[t]he actions and conditions that comprise an event are determine[d] when the event is created.” *Id.* at code (57), 5:49–51, 10:62–64. Nye further discloses that an “event metamodel” may be “instantiated to represent a number of different event models and the corresponding dependencies that interrelate them,” “enabl[ing] the model creator to control what happens to each and every event in the event metamodel without having to modify the underlying structure of the database.” *Id.* at code (57), 5:65–6:1. According to Nye, “[i]nstead of changing the structure of the database, a new data type can be created in the event model” and used immediately, thus “provid[ing] a way to add functionality to a database and immediately make that functionality available for use . . . without having to create new tables or change the relationship that exists between tables.” *Id.* at 6:1–8.

3. *Independent Claim 1*

As reproduced in Section I.D. above, claim 1 recites, in part:

1. A computerized system for manipulating data sets, comprising:

...

[b] a data repository . . . adapted to process, retrieve, and store data contained in the data sets and one or more layers of metadata of the data in the data sets, comprising:

[c] an element module configured to store and uniquely identify a plurality of elements, wherein each of the elements is identified by a unique element identification; [and]

...

[e] a class module configured to define at least one class of the elements and store the class; . . .

Ex. 1001, 16:13–41. Petitioner asserts that a person of ordinary skill would have understood Althoff’s relational database to be a specific

implementation of the data repository as recited in limitation 1[b]. Pet. 14 (citing Ex. 1006, 6:42–55; Ex. 1003 ¶¶ 77–85). According to Petitioner, Althoff’s data repository is adapted to process, retrieve, and store data contained in the data sets, as claimed, because Althoff teaches that it “allows a user to *create, edit and manipulate a user’s object database* (which [Althoff’s] method and system dynamically translates into a set of *relational database structures*),” and Althoff’s relational database stores data in the data sets by translating the user’s object database into a set of relational database structures. *Id.* at 14–15 (quoting Ex. 1006, 3:15–21) (emphasis by Petitioner) (citing Ex. 1006, 3:15–21, 3:25–29; Ex. 1003 ¶¶ 79–80, 84). Petitioner further contends that a person of ordinary skill in the art would understand that Althoff’s relational database is adapted to process, retrieve, and store one or more layers of metadata of the data in the datasets in the same manner as described with respect to the data contained in the data sets. *Id.* (citing Ex. 1006, 3:25–32, 3:41–44; Ex. 1003 ¶¶ 82–83, 85).

With regard to limitation 1[c], Petitioner contends that Althoff teaches representing classes and objects from an object-oriented database within a relational database by explaining that each class in the user’s object database is modeled by a relational database table and that each object of that class is modeled by a row in that table. Pet. 16 (citing Ex. 1006, 3:45–46; Ex. 1003 ¶ 86). Petitioner argues that Althoff describes a system of classes defined in a “class Class” table, citing Althoff’s disclosures that for each of these classes “the system 200 creates a table in the corresponding relational database 203 to correspond to the selected class 101.” *Id.* (citing Ex. 1006, 23:10–12). Petitioner also contends that each table is linked to its particular class in the “class Class” table with a column containing the “class identifier” for each object. *Id.* (citing Ex. 1006, 23:33–34). Petitioner

identifies “objects” in Althoff as corresponding to the recited “elements” in claim 1. Pet. 16 (“Althoff further teaches that these individual class tables *store objects* (*i.e., elements*) . . .”). Petitioner asserts that Althoff’s individual class tables are the claimed “element modules” because each row in the table stores objects (*i.e., elements*). *Id.* (citing Ex. 1006, 23:19–25 (“When an object is populated into the corresponding relational database 203, the system 200 will create a row in the table corresponding to the object’s class 101 [and thus] each object will correspond to one row in each table corresponding to each class 101.”); Ex. 1003 ¶¶ 87–89). Petitioner further contends that Althoff discloses an element module is “configured to uniquely identify a plurality of elements, wherein each of the elements is identified by unique element identification” as claimed because Althoff teaches that for each of the individual class table the first column of the table is the unique identifier (UID) for each object; thus, in each table in which a row is created for the object, the first column corresponds to the object’s UID. *Id.* at 17 (citing Ex. 1006, 23:14–15, 23:26–29; Ex. 1003 ¶ 89).

With respect to limitation 1[e], Petitioner contends that “the *plurality of elements* stored in [Althoff’s] *elements module* are instanced objects of a class.” Pet. 21 (citing Ex. 1006, 3:45–46 (“each class . . . is modeled by a relational database table . . . and each object of that class is modeled by a row in that table.”)). In other words, according to Petitioner, “Althoff discloses that these classes are *classes of the plurality of elements* (for example, a Manufacturer class of the plurality of Manufacturer objects).” *Id.* (citing Ex. 1003 ¶ 95). Petitioner further contends that Althoff teaches that “class modules are defined and stored in a ‘class Class 301’ table (*i.e. class module*)” and that “in the meta-model 220, *each class of the user’s*

object database 100 is modeled by an object of class Class 301.” *Id.* (citing Ex. 1003 ¶ 96; Ex. 1006, 11:13–14). Still further, Petitioner contends that “Althoff further teaches that these ‘*classes*’ of *elements* can be ‘*defined* by a user of the system’ and that ‘Althoff explains that it creates classes (i.e., “class Class” objects) defined by the user by “provid[ing] a capability for the user 201 to select values for the properties 102 of the new object of *class Class 301 which has just been created.*”” *Id.* (quoting Ex. 1006, 1:31–32, 28:28–52 (alteration and emphases by Petitioner)). Finally, according to Petitioner, “Althoff discloses that these classes and their definitions are stored in the ‘class Class 301’ table, where ‘each class . . . is modeled by an object [*i.e.* row] of class Class 301’ and ‘each object of class Class 301 comprises information in table 3-2,” where “[s]uch information includes the ‘name to be displayed for the class’ and a ‘description of the class.’” *Id.* (quoting Ex. 1006, 11:12–16) (citing Ex. 1003 ¶ 97; Ex. 1006, 11:18–48).

Patent Owner raises several arguments in response to Petitioner’s contentions, including that the Petition fails to identify a “data repository” that includes both a “class module” and an “element module” as recited in claim 1. Prelim. Resp. 25–39. Patent Owner points out that “[t]he claims require a ‘data repository’ that has at least eight modules, two of which are an ‘element module’ and a ‘class module,’” but, Patent Owner contends, “[t]he Petition relies on **two different and incompatible databases** in Althoff to find alleged teachings of these modules.” *Id.* at 25.

First, Patent Owner argues, the class description on which Petitioner relies for the recited class module concerns Althoff’s **object-oriented database**, whereas the Petition relies on the **relational database** of Althoff for the alleged “data repository” and its “element module”—consistent with the “modules” of claim 1 being table-based structures that would be found in

a relational database rather than Althoff's object database. Prelim. Resp. 25, 27–29 (citing Pet. 14, 16, 17, 21). According to Patent Owner, Althoff's "two databases are distinct, and Althoff specifically provides a translator to **translate** an object database in a relational database **because** the object database is not compatible with relational databases." *Id.* at 25–26 (citing Ex. 1006, 3:4–24).

Second, Patent Owner argues, "Althoff's relational database structure obviates the need for a 'class module'" by "preserv[ing] . . . class information after translation by creating separate tables for each class created by the user." Prelim. Resp. 26. Thus, Patent Owner contends, "[w]ithin the context of Althoff, a 'class module' would serve no purpose." *Id.*

Third, Patent Owner contends, "the text on which Petitioner relies for the alleged 'class module' in Althoff is not a table or table-based structure," and accordingly, whether under the construction of "module" applied by the Petition or under the "correct construction," the Petition fails to identify a "class module." Prelim. Resp. 26.

Regarding the first point, Patent Owner cites pages 16 and 17 of the Petition and notes that "the Petition clearly maps the 'data repository' of claim 1 onto the relational database of Althoff and maps the 'element module' of claim 1 onto each of the tables of the relational database as depicted in Figure 8 of Althoff." Prelim. Resp. 28–29. Patent Owner contends, however, that "[e]ven assuming this mapping is valid, the Petition nonetheless fails because the Petition fails to identify a 'class module' in the relational database of Althoff (i.e., in the alleged 'data repository' of Althoff)." *Id.* at 29. Rather, Patent Owner points out, "[t]he Petition maps the 'class module' to a class in Althoff's object database." *Id.* In particular,

Patent Owner explains, “[f]or the alleged ‘class module,’ the Petition relies on the discussion of the object-oriented class called ‘Class’ that is used in Althoff’s object database prior to the translation of the object database into a relational database.” *Id.* (citing Pet. 21 (referring to “class Class 301”)). Patent Owner contends that “[t]his class is not a ‘class module’ because it is not a table-based structure,” “[n]or does ‘class Class 301’ interact with the relation database of Althoff.” *Id.* On the contrary, Patent Owner argues, “Althoff provides a database translator precisely because object databases and relational databases are incompatible.” *Id.* (citing Ex. 1006, 2:9–14, 3:4–6).

According to Patent Owner, notwithstanding Petitioner’s contentions, “there is no object named ‘Class’ or ‘Class 301’ in the user’s object database, and no object name ‘Class’ is translated into a ‘class module’ in the relational database.” Prelim. Resp. 32. According to Patent Owner, “‘class Class 301’ is the name of an object-oriented class used by the **object database** of Althoff.” *Id.* at 30 (citing Ex. 1006, 11:12–15). As Patent Owner explains, “[a] class in Althoff is not a structure or a thing, but rather is a data type, much like ‘integer’ and ‘floating point.’” *Id.* (citing Ex. 1006, 1:21–39). For example, Patent Owner posits, “a user may create a telephone number class by creating an object of class Class 301 called ‘Telephone Number,’” such that “the object ‘Telephone Number’ [would be] a member of the class called ‘Class.’” *Id.* However, Patent Owner explains, “this ‘Telephone Number’ object of class ‘Class’ [would] not hold any telephone numbers,” but instead would “just define[] what a telephone number is and

what can be done with it.” *Id.* Figure 1 of Althoff, reproduced below, illustrates Patent Owner’s argument.

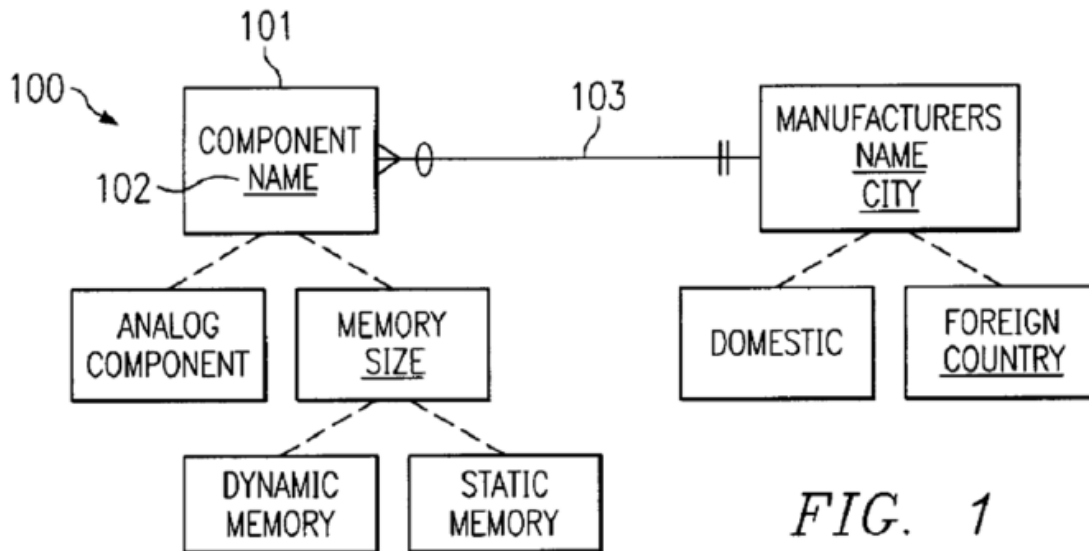


Figure 1 of Althoff, above, “shows a data model of the examples of the user’s object database.” Ex. 1006, 4:59–60. According to Patent Owner, “[i]n the exemplary object database in Althoff as shown in Figure 1 of Althoff, a user has created 8 classes (data types): (1) Component, (2) Analog Component, (3) Memory, (4) Dynamic Memory, (5) Static Memory, (6) Manufacturer, (7) Domestic [Manufacturer], and (8) Foreign [Manufacturer],” and “[e]ach of these classes is created in Althoff’s **object database** by creating an object of class (i.e. of type) Class 301.” Prelim. Resp. 31. For example, Patent Owner contends, “the ‘Component’ object defines what a component is and what can be done with it, and the ‘Manufacturer’ object defines what a Manufacturer is and what can be done with it.” *Id.* With this background, Patent Owner contends, “the Petition’s mapping of ‘class Class 301’ fails for multiple reasons.” *Id.*

First, Patent Owner argues, “‘class Class 301’ is in the **wrong database**,” insofar as “[i]t is defined in the **object** meta-model and is used

Ex. 1006, 5:8–10. Referencing annotated Figure 8, Patent Owner contends that, although “the classes in the user’s object database (objects of type ‘Class’) are converted to ‘element modules’ under the Petition’s mapping,” including “creat[ion of] a ‘Component’ table, ‘Memory’ table, ‘Manufacturer’ table, etc,” no “Class” table is created. *Id.* (citing Pet. 16–17). In fact, Patent Owner argues, “given the design of Althoff, such as [sic] class table (or class module) would be unnecessary” because “Althoff’s relational database preserves the class information by creating separate tables for each class.” *Id.* at 33.

Second, Patent Owner argues, “even if the Petition could combine structures in the object database with structures from the relational database . . . , the Petition is still deficient because the ‘class Class 301’ is not structural,” but is merely “a data type defined by Althoff’s meta-model,” “and thus cannot be a ‘class module.’” Prelim. Resp. 33–35 (citing Ex. 1006, 1:27–31, 11:12–14, Fig. 2). Patent Owner repeats that, “[a]s shown in [Althoff’s] Figure 1, the user creates classes by creating objects that are of the type ‘Class 301,’ such as ‘Component’ or ‘Manufacturer,’” and thus “a user defines classes by created objects of the type ‘Class,’” where each class is defined in a separate object as graphically depicted in Figure 1 of Althoff. *Id.* at 34. Nonetheless, Patent Owner points out, “[t]here is no ‘object’ called ‘Class’ in the user’s object database (or ‘class Class’ or ‘class Class 301,’ or any other variation).” *Id.*; *see also id.* at 32 (“[T]here is no object named ‘Class’ or ‘Class 301’ in the user’s object database, and no object name ‘Class’ is translated into a ‘class module’ in the relational database. To be clear, Petitioner points to nothing from either the user’s object database or the meta-model that is translated into a ‘class

module’ found in the user’s relational database.” (citing Ex. 1006, Figs. 1, 8)).

Finally, noting that at page 21 the Petition maps the claimed class module to a “‘class Class 301’ table” in Althoff, Patent Owner argues that Petitioner “misrepresents Althoff because there is no such ‘table’ in Althoff.” Prelim. Resp. 36–37. According to Patent Owner, “[t]he Petition partially quotes Althoff, 11:12-16 in support, but the Petition radically alters the meaning of the quoted passage via its partial quotation, ellipsis, and insertion.” *Id.* at 37. Althoff’s actual disclosure, Patent Owner argues, makes clear that the cited passage concerns Althoff’s user’s object database and not the relational database, “a detail excised by the Petition’s partial quoting.” *Id.* (citing Ex. 1006, 11:12–16). Moreover, Patent Owner argues, “Althoff nowhere states that class information for each user created object is stored in a ‘row’ of an alleged ‘class Class 301 Table’ as asserted in the Petition.” *Id.* at 38 (citing Pet. 21). According to Patent Owner, the Petition’s reference to Table 3-2 of Althoff as describing information stored in a class Class 301 table is incorrect because Table 3-2 does not concern a row of class table but is instead “merely a descriptive device in the specification of Althoff that describes in a two-column format the information stored within an object of type ‘Class’ in the user’s object database . . . in a form more easily accessible to the reader than a narrative.” *Id.* at 38–39 (citing Pet. 21; Ex. 1006, 11:15–47).

We agree with Patent Owner that the Petition fails to show that the cited prior art teaches or suggests the recited “class module” of claim 1. As Patent Owner points out, claim 1 recites a “data repository” that includes at least eight modules, two of which are an “element module” (limitation 1[c]) and a “class module” (limitation 1[e]). *See supra* § I.D (reproducing claim 1

with paragraph designations used in the Petition); Prelim. Resp. 25.

Petitioner relies on Althoff's relational database as teaching the recited data repository and, *inter alia*, the recited element module. *See* Pet. 14–17.

However, Petitioner relies on Althoff's "class Class 301," a class used by Althoff's object-oriented database, as the basis for its arguments with respect to the recited class module. Pet. 21. Although Althoff provides a database translator for translation between object oriented and relational database structures (*see, e.g.*, Ex. 1006, 3:4–6 (stating that "it would be advantageous to provide a method of compilation and translation between object oriented and relational database structures")), we agree with Patent Owner that there is no object named "Class" or "Class 301" in the user's object database in Althoff and that no object name "Class" is translated into a "class module" in the relational database. Prelim. Resp. 32. We further agree with Patent Owner that Petitioner points to nothing from either the user's object database or the meta-model that is translated into a "class module" found in the user's relational database. *Id.* As Patent Owner argues, class Class 301 in Althoff is not an object in the user's object database, but instead is a class (i.e., a data type) defined by the meta-model that can be used by the user's object database as a data type when the user creates objects. *Id.* at 29–30, 32, 37–38. Still further, we agree with Patent Owner that Althoff teaches that relational databases and object databases are distinct and incompatible, a problem that Althoff seeks to address with its disclosed systems and methods. *See* Prelim. Resp. 25–26, 29 (citing Ex. 1006, 2:9–14, 3:4–24).

Patent Owner's contentions are also persuasive that there is neither any "class Class 301" table disclosed in Althoff, notwithstanding Petitioner's assertions, nor any reason to generate such a table. *See* Prelim. Resp. 36–39 (citing Pet. 21). As Patent Owner notes, by creating separate

tables for each class, e.g., a Component table and a Manufacturer table, Althoff preserves class information. *Id.* at 33. But Althoff does not state that class information for each user-created object is stored in a “row” of a “class Class 301 table” as asserted in the Petition, and we agree with Patent Owner that Althoff would have no need to describe a translation process to convert objects into tables if the classes of the object database were already stored “rows” in a “table.” *Id.* at 38. With regard to the Petition’s reference to Althoff’s disclosure that “each object of class Class 301 comprises information in table 3-2” (Pet. 21 (quoting Ex. 1006, 11:12–16)), we agree with Patent Owner that Table 3-2 of Althoff does not concern a row of a “class table”; instead, as Patent Owner explains, “Table 3-2 is merely a descriptive device in the specification of Althoff that describes in a two-column format the information stored within an object of type ‘Class’ in the user’s object database . . . in a form more easily accessible to the reader than a narrative.” Prelim. Resp. 38–39 (citing Ex. 1006, 11:15–47).

For at least the foregoing reasons, we conclude that Petitioner has not shown adequately at this stage of the proceeding that Althoff teaches or suggests limitation 1[e]. Moreover, Petitioner does not allege that Nye addresses the above-identified deficiencies of Althoff with respect to that limitation. Thus, we determine that Petitioner has not demonstrated a reasonable likelihood it will succeed in its challenge to claim 1. We need not address the remaining limitations of claim 1 or Patent Owner’s arguments as to why those limitations have not been sufficiently shown. *See* Prelim. Resp. 39–50.

4. *Independent Claims 14 and 27*

Claim 14 is drawn to a method. Claim 27 is drawn to a non-transitory computer medium. Petitioner relies for claims 14 and 27 on its arguments

with respect to claim 1, merely citing for the limitations of claims 14 and 27 corresponding limitations of claim 1 and arguing that claims 14 and 27 are unpatentable for the same reasons as alleged for claim 1. Pet. 49–51. For the reasons discussed above with respect to claim 1, we determine that Petitioner also has not demonstrated a reasonable likelihood it would succeed in its challenge to claims 14 and 27.

5. *Dependent Claims 2–9, 11–13, 15–22, 24–26, 28–35, and 37–39*

Claims 2–9 and 11–13 depend from claim 1. Claims 15–22 and 24–26 depend from claim 14. Claims 28–35 and 37–39 depend from claim 27. Having failed to demonstrate a reasonable likelihood of prevailing for independent claims 1, 14, and 27, Petitioner also fails to demonstrate a reasonable likelihood of prevailing for these dependent claims.

E. *Obviousness of Claims 10, 23, and 36 over Althoff, Nye, and Surjanto*

Petitioner relies on Surjanto as disclosing, for example, XML document management extensions “specifically designed to extend [object-oriented] databases like Althoff’s to provide the known benefits of XML document support.” Pet. 52 (citing Ex. 1003 ¶ 158; Ex. 1008, 9); *see also id.* at 5–6, 51–57.

Claim 10, 23, and 36 depend from independent claims 1, 14, and 27, respectively. Petitioner has not identified disclosure in Surjanto that teaches the subject matter of the independent claims lacking from Althoff and Nye. Therefore, we determine that Petitioner has not demonstrated a reasonable likelihood it would succeed in its challenge to claims 10, 23, and 36 as unpatentable over Althoff, Nye, and Surjanto.

F. Obviousness of Claims 6, 7, 19, 20, 32, and 33 over Althoff, Nye, and Moore

Petitioner relies on Moore as providing additional disclosure of the “named text module” and “tuple module” of claims 6, 7, 19, 20, 32, and 33, in the event that “Patent Owner may argue that the combination of Althoff and Nye does not disclose” those modules. Pet. 57–58; *see also id.* at 9, 58–68.

As stated above, claims 6 and 7 depend from independent claim 1; claims 19 and 20 depend from independent claim 14; and claims 32 and 33 depend from independent claim 27. Petitioner has not identified disclosure in Moore that teaches the subject matter of the independent claims lacking from Althoff and Nye. Therefore, we determine that Petitioner also has not demonstrated a reasonable likelihood it would succeed in its challenge to claims 6, 7, 19, 20, 32, and 33 as unpatentable over Althoff, Nye, and Moore.

III. CONCLUSION

After considering the evidence and arguments presented in the Petition and the Preliminary Response, we determine that Petitioner does not demonstrate a reasonable likelihood of prevailing with respect to any of the challenged claims, and thus, we do not institute an *inter partes* review.

IV. ORDER

In consideration of the foregoing, it is hereby

ORDERED that the Petition is *denied* as to all challenged claims, and no trial is instituted.

IPR2023-01071
Patent 7,979,468 B2

FOR PETITIONER:

James M. Glass
John T. McKee
Quincy Lu
QUINN EMANUEL URQUHART & SULLIVAN, LLP
jimglass@quinnemanuel.com
johnmckee@quinnemanuel.com
quincylu@quinnemanuel.com

FOR PATENT OWNER:

Steven R. Katz
Andrew G. Pearson
Jason W. Wolff
FISH & RICHARDSON P.C.
katz@fr.com
pearson@fr.com
wolff@fr.com