

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

ORACLE CORPORATION,
Petitioner,

v.

THOUGHT, INC.,
Patent Owner.

Case IPR2014-00118
Patent 7,103,600 B2

Before KARL D. EASTHOM, BARBARA A. BENOIT, and
STACEY G. WHITE, *Administrative Patent Judges*.

BENOIT, *Administrative Patent Judge*.

FINAL WRITTEN DECISION
35 U.S.C. § 318(a) and 37 C.F.R. § 42.73

I. INTRODUCTION

We have jurisdiction to hear this *inter partes* review under 35 U.S.C. § 6(c). This Final Written Decision is issued pursuant to 35 U.S.C. § 318(a) and 37 C.F.R. § 42.73. For the reasons that follow, we determine that Petitioner has shown by a preponderance of the evidence that claims 1, 2, 6, and 8 of U.S. Patent No. 7,103,600 B2 (Ex. 1001; “the ’600 patent”) are unpatentable.

A. Procedural History

Oracle Corporation (“Petitioner”) filed a corrected Petition (“Pet.”; Paper 5) for an *inter partes* review of claims 1–17 of the ’600 patent. Patent Owner, Thought, Inc., did not file a preliminary response opposing institution of a review. On April 25, 2014, pursuant to 35 U.S.C. § 314(a), we instituted an *inter partes* review for claims 1, 2, 6, and 8¹ of the ’600 patent as unpatentable under 35 U.S.C. § 103(a) for obviousness over Demuth² and under 35 U.S.C. § 103(a) for obviousness over Demuth and Kurniawan.³ Paper 16 (“Inst. Dec.”).

Subsequent to institution, Patent Owner filed a Patent Owner Response (Paper 29; “PO Resp.”), and Petitioner filed a Reply (Paper 34; “Reply”).

¹ Prior to institution, Petitioner filed an unopposed motion to withdraw its alleged grounds of unpatentability against all claims except 1, 2, 6, and 8. Paper 14. We granted Petitioner’s motion and therefore, we considered only challenges leveled against claims 1, 2, 6, and 8. Inst. Dec. 3–4.

² U.S. Patent Pub. No. 2003/0212987, published Nov. 13, 2003, filed on Feb. 28, 2002 (Ex. 1006) (“Demuth”).

³ BUDI KURNIAWAN, JAVA FOR THE WEB WITH SERVLETS, JSP, AND EJB, Apr. 2002 (Ex. 1009) (“Kurniawan”).

Patent Owner filed observations on the cross-examination of Petitioner's declarant (Paper 37), to which Petitioner filed a response (Paper 40).

An oral hearing was held on November 18, 2014.⁴

B. Related Matters

Petitioner represents that the '600 patent was asserted against it in *Thought, Inc. v. Oracle Corp.*, No. 3:12-05601 (N.D. Cal.). Pet. 1; see Paper 8, 1 (Patent Owner's Mandatory Notice). Petitioner also represents that the district court proceeding includes another patent related⁵ to the '600 patent and for which Petitioner also requested *inter partes* review—U.S. Patent No. 7,167,862 (Case IPR2014-00119). Pet. 1.

C. The '600 Patent

The '600 patent, titled "Displayable Presentation Page and SQL Searchable Relational Data Source Implementation of a System, Method and Software for Creating or Maintaining Distributed Transparent Persistence of Complex Data Objects and Their Data Relationships," issued on September 5, 2006, from an application filed March 6, 2003. It makes no priority claim. The '600 patent generally relates to storing and accessing data used by a computer system according to a particular computer programming paradigm—object-

⁴ This proceeding and IPR2014-00119 involve the same parties and some similar issues. The oral arguments for both reviews were merged and conducted at the same time. A transcript of the oral hearing is included in the record as Paper 42.

⁵ Neither the '600 patent challenged here nor U.S. Patent No. 7,167,862 B2 challenged in IPR2014-00119 make a priority claim. Petitioner represents the patents overlap in subject matter and claim language.

oriented programming.⁶ Ex. 1001, col. 1, ll. 12–14; *see id.* at col. 1, ll. 30–31; col. 4, ll. 1–18. Petitioner, supported by the testimony of Dr. Antony L. Hosking as set forth in his declaration (Ex. 1012), explains that some computer systems use the notion of “objects” to help a computer programmer develop a computer system. Pet. 6 (citing Ex. 1012 ¶ 7.1.2). As an illustration, Petitioner describes a flight reservation system that has objects representing airports, planes, and seats on particular flights. *Id.* This aids computer programmers in developing a computer system, because programmers are able to think “in terms of the properties and functions of a relevant object (e.g., does this plane, for this flight, have any unsold seats),” rather than being concerned with “lower-level technical details about how the program works.” *Id.* (citing Ex. 1012 ¶ 7.1.2). Patent Owner’s declarant Sam Malek, Ph.D. similarly testifies regarding objects in object-oriented programming. Ex. 2001 ¶ 23; *see also* PO Resp. 10–11 (summary of the ’600 patent).

Specifically, the ’600 patent relates to “a system, methods and software for creating or maintaining distributed transparent persistence of complex data objects and associated data stores.” Ex. 1001, col. 1, ll. 15–18. The ’600 patent addresses the problem of “persisting” data used by object-oriented programs in a distributed computer environment in which information needed to persist, or

⁶ MICROSOFT COMPUTER DICTIONARY 373 (5th ed. 2002) (defining “object-oriented programming” as “[a] programming paradigm in which a program is viewed as a collection of discrete objects that are self-contained collections of data structures and routines that interact with other objects”); *see also* Ex. 1012 ¶ 4.1 (Declaration of Antony L. Hosking, Ph.D.); Ex. 2001 ¶ 23 (Declaration of Sam Malek, Ph.D.).

store, the data objects may be found in different computer systems. *See id.* at col. 1, ll. 47–52.

Figure 3 of the '600 patent shows a preferred method for addressing this problem. *See id.* at col. 9, ll. 48–49. Figure 3 of the '600 patent is set forth below:

FIGURE 3

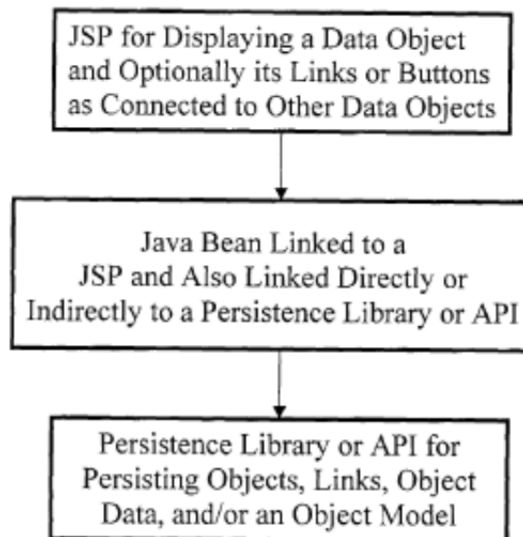


Figure 3 shows a programming flow chart for creating, displaying, updating, and persisting data and/or links of a complex data object.

Id. at col. 9, ll. 48–52; Fig. 3. Three programming modules are shown in Figure 3. *Id.* at Fig. 3; col. 9, l. 48–col. 10, l. 37. The first programming module uses “a displayable presentation page having embedded object programming code, such as a Java Server Page (JSP),” to display the data of a data object and, optionally, names of links to show how the data object is related to other data objects in “a [complex data object] graph” or model. *Id.* at

col. 9, ll. 48–63. According to the '600 patent, an example of a complex data object is a customer object and its related objects of billing address, orders, and items ordered. *Id.* at col. 9, ll. 4–24; Fig. 1.

The second programming module is a “programming object,” such as a “Java Bean,” that is linked, through the first programming module, to the displayed data object. *Id.* at col. 10, ll. 10–14. The second programming module also is linked to a third programming module that includes “programming logic for tracking and persisting” to data source changes to the data object. *Id.* at col. 10, ll. 10–32.

D. Illustrative Claim of the '600 Patent

Claim 1 of the '600 patent is the only challenged independent claim and recites the following:

1. A system for creating or maintaining transparent persistence of a member selected from the group consisting of a data object, an object graph model and a portion of an object graph model when a user of the system is creating, maintaining, accessing or navigating complex data objects as a complex data object graph model, comprising:

a) a set of definitions for the relationships between a data source schema and objects capable of storing data for an object language application, wherein the set of definitions is stored in a repository;

b) a set of definitions for the relationships between objects for an object language application, wherein the set of definitions is part of an object application navigation model;

c) a list of objects, or a set of objects that are to be persisted, wherein the list of objects or set of objects is part of an object application navigation model;

d) an object or set of objects as part of a computer implemented method that contains the computer logic capable of persisting an indicated object or set of objects;

e) an input method to provide the computer logic of d) with data corresponding to the location of information related to a), b) and c), and

f) at least one data source to in which persisted data may be stored.

Id. at col. 38, ll. 39–65.

II. ANALYSIS

A. Claim Construction

In an *inter partes* review, claim terms in an unexpired patent are interpreted according to their broadest reasonable construction in light of the specification of the patent in which they appear. 37 C.F.R. § 42.100(b); see Office Patent Trial Practice Guide, 77 Fed. Reg. 48,756, 48,766 (Aug. 14, 2012); *In re Cuozzo Speed Techs., LLC*, 778 F.3d 1271, 1279–82 (Fed. Cir. 2015). Under the broadest reasonable construction standard, claim terms are presumed to be given their ordinary and customary meaning as would be understood by one of ordinary skill in the art in the context of the entire disclosure. *In re Translogic Tech., Inc.*, 504 F.3d 1249, 1257 (Fed. Cir. 2007). An inventor may provide a meaning for a term that is different from its ordinary meaning by defining the term in the specification with reasonable clarity, deliberateness, and precision. *In re Paulsen*, 30 F.3d 1475, 1480 (Fed. Cir. 1994).

We construe the claim language below in accordance with these principles. No other terms require express construction.

1. “*transparent persistence*”

A central dispute between the parties concerns “transparent persistence,” despite the fact that none of the elements in the body of any of the challenged claims recite that term. Although the inventor did not provide a definition of transparent persistence, the inventor provided a definition of *distributed* transparent persistence in the Specification of the ’600 patent. See Ex. 1001, col. 3, ll. 48–54 (indicating each defined term is to be given the meaning ascribed to it in the list of definitions), col. 5, ll. 45–51 (defining distributed transparent persistence). Distributed transparent persistence is defined as “a term employed herein as an abstraction referring to the concept of providing persistence for a member selected from the group consisting of a data object, a data object graph, associated data and data object relationships in a *distributed environment* without the need for the insertion of byte code or data objects in an object model or schema.” Ex. 1001, col. 5, ll. 45–51 (emphasis added). The inventor provided a meaning for distributed transparent persistence with sufficient clarity, deliberateness, and precision to define the term. See *Paulsen*, 30 F.3d at 1480. Other than being required to be within a distributed environment, it follows that transparent persistence would mean the same thing as distributed transparent persistence.

As Patent Owner indicates, the Specification of the ’600 patent consistently references the invention as providing transparent persistence—and often, distributed transparent persistence—including in the title of the ’600 patent, the “Field of the Invention” section, the “Summary of the Invention” section, and the Abstract. See Ex. 1001, Title, col. 1, ll. 15–18 (stating “the

present invention relates to a system, methods and software for creating or maintaining distributed transparent persistence of complex data objects and associated data stores”), col. 6, ll. 61–64 (stating “[a]n object of the present invention is to provide a system for creating or maintaining transparent persistence of a complex data object” or graph), Abstract (indicating the “invention provides systems, methods and software for creating or maintaining distributed transparent persistence of complex data objects and associated data stores”).

On this basis, Patent Owner contends that the recitation of “transparent persistence” in the preamble is a limiting element of the claims. PO Resp. 14–15. Petitioner disagrees. Reply 1–2.

“In general, a preamble limits the invention if it recites essential structure or steps, or if it is ‘necessary to give life, meaning, and vitality’ to the claim.” *Catalina Mktg. Int’l, Inc. v. Coolsavings.com, Inc.*, 289 F.3d 801, 808 (Fed. Cir. 2002). A preamble, however, “generally is not limiting when the claim body describes a structurally complete invention such that deletion of the preamble phrase does not affect the structure or steps of the claimed invention.” *Id.* at 809. One guidepost for determining the effect of a preamble on claim scope is whether the preamble language provides antecedent basis for any limitation in the body of the claim. *Id.* at 808. Moreover, a preamble describing the purpose or intended use of an invention generally does not limit the claim. *Id.* at 809.

The elements recited in the body of claim 1—three sets of information in elements (a), (b), and (c); a computer-implemented method in element (d); an input method in element (e); and a data source in element (f)—do not rely on

the preamble language as an antecedent basis for any limitation in the body of the claim. Few of the recited elements of the system use antecedent basis to create connections between the recited elements. For example, none of the information recited in elements (a), (b), and (c) use antecedent basis to describe relationships with other information recited in the elements (a), (b), or (c), or use antecedent basis to describe relationships with the object (or set of objects) to be persisted, as recited in element (d). Nor does claim 1 require any of the objects in the list of objects (or set of objects) to be persisted (recited in element (c)), to be related to any of the objects capable of storing data identified in the definitions for relationships (recited in element (a)) or the data source to which persisted data may be stored (recited in element (f)).

Further, claim 1 does not require the object (or set of objects) as part of a computer-implemented method that contains the computer logic capable of persisting an indicated object or set of objects (as recited in element d), to be capable of persisting a particular type of object, such as an object capable of storing data for an object language application (as recited in element (a)). In another example, the object application navigation models recited in elements (b) and (c) are not necessarily the same object application navigation model because elements (b) and (c) each recite “*an* object application navigation model” (emphasis added). The input method recited in element (e), however, provides the computer logic of element (d) with the locations of the information recited in other elements of claim 1. Even so, because claim 1 does not define relationships among many of the information and objects recited in

the claim, the scope of claim 1 is broader than it would be if the claim recited such relationships.

Turning again to whether the preamble is limiting, the intended use of the invention “for creating or maintaining transparent persistence of a member selected from the group consisting of a data object, an object graph model and a portion of an object graph model when a user of the system is creating, maintaining, accessing or navigating complex data objects as a complex data object graph model” does not recite essential structure to complete the invention recited in elements (a)–(f). *See Pitney Bowes, Inc. v. Hewlett-Packard Co.*, 182 F.3d 1298, 1305 (Fed. Cir. 1999) (if the body of the claim fully and intrinsically sets forth all the limitations of the claimed invention, and the preamble merely states, for example, the purpose or intended use of the invention, then the preamble is not considered a limitation and is of no significance to claim construction). Rather, the preamble describes an intended use (i.e., “creating or maintaining transparent persistence”) under certain conditions (i.e., “when a user of the system is creating, maintaining, accessing or navigating complex data objects as a complex data object graph model”).

The centrality of transparent persistence to the invention provides evidence that the scope of the claims as a whole relates to transparent persistence. *Cf. Trading Techs. Int’l, Inc. v. eSpeed, Inc.*, 595 F.3d 1340, 1353–54 (Fed. Cir. 2010) (concluding that specification’s “reference to ‘the present invention’ strongly suggests” that the claim carries the same meaning). Moreover, claims in an unexpired patent are construed according to their

broadest reasonable construction *in light of the specification* of the patent in which they appear. 37 C.F.R. § 42.100(b); *Cuozzo*, 778 F.3d at 1279–82.

Although the elements in the body of claim 1 do not recite expressly “transparent persistence,” claim 1 requires storing, in a repository, definitions for the relationships between a data source schema and objects capable of storing data, which is “an essential requirement for achieving transparent persistence,” according to Patent Owner’s declarant Dr. Malek (Ex. 2001 ¶ 48); *see* Ex. 1001, col. 38, ll. 45–48 (claim 1, element (a)). Second, claim 1 twice recites an object application navigation model, which is another important element in transparent persistence. *See* Ex. 2001 ¶ 49 (Dr. Malek testifying that the invention discloses an object application navigation model used to determine the list of objects that should be persisted to a database); Ex. 1001, col. 38, ll. 52–53, 56 (claim 1, elements (b), (c)). Because aspects of transparent persistence are recited expressly in the body of claim 1, the concept of transparent persistence is present in the claim body; therefore, the preamble is not “necessary to give life, meaning, and vitality” to the claim.

Moreover, the general concept of creating or maintaining transparent persistence was known at least as early as November 2001, sixteen months before the effective filing date of the application that issued as the ’600 patent. An article by Sun Microsystems described one of Patent Owner’s CocoBase products in November 2001 as providing “Dynamic Transparent Persistence” to allow a software programmer “to persist relational data with a simple, non-invasive . . . object” and “[w]ithout any byte code or object model intrusion.” Ex. 1016, 2. The article further indicates that “the object is unaware

of the data structure used to persist it.” *Id.* This further supports our determination that the preamble is not “necessary to give life, meaning, and vitality” to the claim.

Reading the claim as a whole and applying the broadest reasonable construction of the claim language in light of the Specification, we conclude that the “creating or maintaining transparent persistence” language in the preamble of claim 1 is not a limitation. Accordingly, we hold that claim 1 does not preclude the insertion of byte code or data objects in an object model or schema, which are features of transparent persistence. Notably, claims 2 and 8, which depend from claim 1, each additionally recite that the system “does not require any modifications to an object model or the inclusion of any persistence byte code in the object model in order to provide persistence for all or a portion of the complex data object graph model.”

2. “object”

Independent claim 1 recites “objects capable of storing data for an object language application,” “objects for an object language application,” “objects that are to be persisted,” “objects [that are] part of an object application navigation model,” and “objects [that are] part of [a certain type of] a computer implemented method.” The ’600 patent sets forth a definition for an “object”:

“Object” as used in the context of this document is a general term referring to a logic unit in a computer application or a computer software program where the application or program is based upon an object[-]oriented programming language (e.g., Java). The term “object” may ordinarily be used interchangeably with the term “class” as a template or as an instance depending on the context.

Ex. 1001, col. 4, ll. 1–7; *see also id.* at col. 3, l. 46–col. 6, l. 57 (section titled “DEFINITIONS”). Dr. Hosking explains the difference between an object class and an object instance in an object-oriented application for a flight reservation system:

Using object-oriented programming languages, programmers create[] “classes[,]” which are templates for real world entities such as “planes” or “people.” From these classes, programmers could create particular “objects,” such as “a plane with serial number ‘123’ belonging to [a particular] airline”. . . or “a person with a [particular name] and [telephone number].”

[A] plane class may define . . . all instances of that class (the various plane objects)

Ex. 1012 ¶¶ 4.1–4.2.

Based on the definition in the Specification and supported by Dr. Hosking’s testimony, Petitioner proposes that the term “object” should be construed as “either a template or an instance.” Pet. 8. Although we agree that an object, by its definition in the Specification, can be a class as a template or an instance, Petitioner’s proposed construction places insufficient weight on other aspects of the definition of “object” in the Specification.

For these reasons, in the Decision to Institute, we construed “object” as “a logic unit in a computer application or a computer software program based on an object-oriented programming language (e.g., Java) in which, depending on context, the logic unit refers either to an instance or to a class as a template.” Inst. Dec. 11. Patent Owner agrees with this construction. PO Resp. 12. Petitioner does not challenge this construction. *See generally* Reply 1–3.

Having considered whether the construction set forth in the Decision to Institute should be changed in light of evidence introduced during trial, we are not persuaded that any modification is necessary. Therefore, we maintain the construction of “object” as “a logic unit in a computer application or a computer software program based on an object-oriented programming language (e.g., Java) in which, depending on context, the logic unit refers either to an instance or to a class as a template.”

3. “*data source schema*”

Independent claim 1 recites “a set of definitions for the relationships between a data source schema and objects capable of storing data for an object language application.” The ’600 patent does not use the term “data source schema” other than in the claims and in repeating substantially the claim language in the Description of the Invention section. *See* Ex. 1001, col. 10, ll. 55–58; col. 12, ll. 46–51, 62–66. Nor does the ’600 patent set forth a definition of “data source schema.” The ’600 patent, however, does set forth a definition for an “object schema”:

“Object schema” is a term employed herein as an abstraction referring to the set of data object classes that describe the possible data objects that can be created, modified or maintained in an application, or describing an instance of a set of data object classes in an application.

Ex. 1001, col. 5, ll. 40–44. The ordinary meaning of the term “schema” is “[a] logical description of the data in a database, including definitions and relationships of data.”⁷

In the Decision to Institute, we construed “a data source schema,” in light of the Specification and the ordinary meaning of the term “schema,” as “an abstraction describing data in a data base used as a source of data in an application.” Dec. Inst. 12. Patent Owner agrees with this construction. PO Resp. 17. Petitioner does not challenge this construction. *See generally* Reply 1–3.

Having considered whether the construction set forth in the Decision to Institute should be changed in light of evidence introduced during trial, we are not persuaded that any modification is necessary. Therefore, we maintain the construction of “a data source schema” as “an abstraction describing data in a database used as a source of data in an application.”

4. “*object application navigation model*”

Independent claim 1 recites “the set of definitions is part of an object application navigation model” in element (b) and “the list of objects or set of objects is part of an object application navigation model” in element (c). Thus, claim 1 only indicates expressly an object application navigation model includes information specified in elements (b) and (c). Further, by reciting “an object application navigation model” in each element (b) and (c), claim 1 does

⁷ MCGRAW-HILL DICTIONARY OF SCIENTIFIC AND TECHNICAL TERMS 1864 (6th ed. 2003).

not require the set of definitions in element (b) and the list of objects or set of objects in element (c) necessarily are part of the same object application navigation model.

The '600 patent does not define the precise term “object application navigation model.” The Specification, however, defines a “navigation model”:

a special type of application model that is applied specifically to a description (or other representation) of how objects can relate to each other and what might be the expected behavior when a [complex data object graph] is navigated for a certain purpose.

Ex. 1001, col. 5, ll. 35-39. According to the '600 patent, a “complex data object graph” is “an abstraction to logically represent a set of complex data objects and a set of their corresponding relationships.” *Id.* at col. 4, ll. 62–65.

In turn, the '600 patent defines a “complex data object”:

a data object that has at least one or more relationships with itself, or at least one or more relationships with one or more other data object(s). In a given instance of a [complex data object] at least one relationship is populated as a link, as defined below. A [complex data object] may have a multiplicity of different relationships with itself or with one or more additional [complex data objects].

Id. at col. 4, ll. 19–26. The '600 patent, in turn, defines “link” in the context of a complex data object:

identif[ying] a particular occurrence of a relationship between a [complex data object] and itself, between a [complex data object] and another [complex data object]. The occurrence of at least one populated link results in an instance of a [complex data object]. This may be considered as a synonym for a ‘relationship’ between two objects.

Id. at col. 4, ll. 46–51.

According to the '600 patent, then, a “navigation model” is a model describing how objects relate to one another and the expected behavior when a complex data object graph is navigated for a certain purpose; “a complex data object graph” is an abstraction of an object that has one or more relationships with itself or another object; and a “relationship” is populated as a link between two objects or an object with itself.

For these reasons, in the Decision to Institute, we construed an “object application navigation model” as “a model describing how objects in an application relate to themselves or one another, and the expected behavior when relationships, populated as a link between the objects or as a link with itself, are navigated for a certain purpose.” Dec. Inst. 14.

Petitioner does not challenge this construction. *See generally* Reply 1–3. Based on Dr. Malek’s testimony, Patent Owner, although agreeing that this construction is correct, further contends that an object application navigation model also is distinct from a data object graph that “shows data objects as the edges and relationships as the nodes.” PO Resp. 16 (citing Ex. 2001 ¶¶ 67, 68, 70). Patent Owner contends that claim 1 precludes an object application navigation model to be a data object graph because claim 1 recites “the expected behavior when relationships . . . are navigated for a certain purpose.” *Id.*

Contrary to Patent Owner’s arguments, an object application navigation model does not have to be distinct from a graph that shows objects as the edges and relationships as the nodes and that holds application data. *See* PO Resp. 26–27 (arguing a graph of the data objects that hold application data is not an

object application navigation model); Ex. 2001 ¶ 68 (testifying “the object application navigation model is distinct from a graph of data objects that hold the data within an application.”). A navigation model by the Specification’s express definition is a model describing how objects relate to one another. A graph showing objects as the edges and relationships as the nodes is one way to describe how objects relate to one another and, therefore, should not be excluded. The express definition does not preclude objects that hold application data.

Moreover, Patent Owner’s contention does not explain sufficiently how an “*object application* navigation model” recited in the claims (emphasis added) is different from the “navigation model” defined in the Specification (Ex. 1001, col. 5, ll. 35–49). Patent Owner’s contention does not place sufficient weight on “object application” in the term “object application navigation model.” Moreover, claim 1 itself recites “a set of objects is part of an object application navigation model,” which indicates objects may be included as part of an object application navigation model. In requiring a set of objects, claim 1 expressly allows object instances to be part of an object application navigation model, in view of the construction of “object” as referring either to an instance or a class depending on context, which further undermines Patent Owner’s contentions that an object application navigation model is distinct from “a graph of data objects that hold data within an application.” See § II.A.2 (construction of object).

Nor are we persuaded by Patent Owner’s reliance on examples from the Specification that disclose multiple navigation models may be provided for the

same complex data object graph (PO Resp. 16 (citing Ex. 1001, col. 14, ll. 18–25, col. 15, ll. 17–18)). The Specification’s examples are from particular implementations—the CDOG Navigator API and the CocoNavigator API. *See* Ex. 1001, col. 14, ll. 18–20 (“Each CDOG (or CDOG model) managed by the CDOG Navigator API can be associated by the CDOG Navigator API with a CDOG descriptor (such as a file) that may be utilized to represent all or part of a ‘navigation model’.”); *id.* at col. 15, ll. 13–23 (“Some preferred features provided by the CocoNavigator API . . . are as follows: . . . [a] preferred embodiment of the CocoNavigator API or an associated program module is configured to allow many navigation models to be used with the same set of java classes and CocoBase maps that are associated with a CDOG model.”).

We decline to read limitations into a claim even from a preferred embodiment described in the Specification. *See In re Am. Acad. of Sci. Tech Ctr.*, 367 F.3d 1359, 1364 (Fed. Cir. 2004). Moreover, the examples refer to “the navigation model,” rather than the recited “object application navigation model.”

Weighing Dr. Malek’s testimony supporting Patent Owner’s contention that an object application navigation model also is distinct from a data object graph that “shows data objects as the edges and relationships as the nodes” (PO Resp. 16 (citing Ex. 2001 ¶¶ 67, 68, 70)) against the evidence of the written description and claim 1, we do not agree that an object application navigation model necessarily must be separate from a data object graph. It is within our discretion to assign the appropriate weight to the testimony offered by Dr. Malek. *See, e.g., Yorkey v. Diab*, 601 F.3d 1279, 1284 (Fed. Cir. 2010)

(holding the Board has discretion to give more weight to one item of evidence over another “unless no reasonable trier of fact could have done so”); *Am. Acad. of Sci. Tech Ctr.*, 367 F.3d at 1368 (“[T]he Board is entitled to weigh the declarations and conclude that the lack of factual corroboration warrants discounting the opinions expressed in the declarations.”).

Thus, we have insufficient evidence before us that one of ordinary skill in the art would consider “an object application navigation model” to be distinct from a data object graph. Likewise, neither claim 1 nor the written description of the ’600 patent clearly requires “an object application navigation model” to be distinct from a data object graph. Accordingly, we maintain the construction of an “object application navigation model” as “a model describing how objects in an application relate to themselves or one another, and the expected behavior when relationships, populated as a link between the objects or as a link with itself, are navigated for a certain purpose.”

5. “byte code”

Claim 2, which depends from claim 1, recites “does not require . . . the inclusion of any persistence byte code in the object model.” The ’600 patent does not set forth a definition of the term “byte code.”

One of ordinary skill in the art at the time of the invention would have understood the term “byte code” to mean compiled Java programs that can be transferred across a network and executed by the Java virtual machine.⁸ The

⁸ MCGRAW-HILL DICTIONARY OF SCIENTIFIC AND TECHNICAL TERMS 305 (6th ed. 2003) (defining “byte code”).

'600 patent's use of "byte code" is consistent with its ordinary meaning. For example, the '600 patent indicates an aspect of the invention is an object model that does not include a certain type of "byte code." See Ex. 1001, col. 1, ll. 43–47; col. 3, ll. 23–29; col. 5, ll. 45–51 (distributed transparent persistence of data objects provided without inserting any byte code); col. 20, ll. 11–13 (distributed transparent persistence of data objects provided without byte code modification).

For these reasons, in the Decision to Institute, we construed "byte code" as "a compiled Java program that can be transferred across a network and executed by the Java virtual machine." Dec. Inst. 14–15. Patent Owner agrees with this construction. PO Resp. 12. Petitioner does not challenge this construction. See *generally* Reply 1–3. Having considered whether the construction set forth in the Decision to Institute should be changed in light of evidence introduced during trial, we are not persuaded that any modification is necessary. Therefore, we maintain the construction of "byte code" as "a compiled Java program that can be transferred across a network and executed by the Java virtual machine."

6. "optionally"

Claim 6, which depends from independent claim 1, additionally recites "providing a system for displaying, updating or creating the data or structure of one or more data objects, and optionally its links to other objects of an object model, by utilizing a displayable presentation page format . . . and persisting" various types of data, among other limitations. In our Decision to Institute, we construed "optionally its links to other objects of an object model" as a feature

that could be included in the claimed system but is not included necessarily in the claimed system. Dec. Inst. 15–16. Neither party disagrees with this construction, nor do we need to change our construction in light of any evidence introduced during trial. Therefore, we maintain this claim construction.

B. Principles of Law

To prevail in challenging claims 1, 2, 6, and 8 of the '600 patent, Petitioner must demonstrate by a preponderance of the evidence that the claims are unpatentable. 35 U.S.C. § 316(e); 37 C.F.R. § 42.1(d). A claim is unpatentable under 35 U.S.C. § 103(a) if the differences between the claimed subject matter and the prior art are such that the subject matter, as a whole, would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. *KSR Int'l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007). The question of obviousness is resolved on the basis of underlying factual determinations including the following: (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 nonobviousness. *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966).

C. Level of Ordinary Skill in the Art

The level of ordinary skill in the art is relevant to the governing standards we apply in the rest of our analysis. The person of ordinary skill in the art is a hypothetical person who is presumed to have known the relevant art at the time

of the invention. *In re GPAC, Inc.*, 57 F.3d 1573, 1579 (Fed. Cir. 1995). Factors that may be considered in determining the level of ordinary skill in the art include, but are not limited to, the types of problems encountered in the art, the sophistication of the technology, and educational level of active workers in the field. *Id.* In a given case, one or more factors may predominate. *Id.*

Relying on the testimony of Dr. Hosking, Petitioner contends a person of ordinary skill in the art would have been a person with a Master of Science degree in computer science, electrical engineering, or a related field, and at least three years of working experience in software development, database design, or a related technology. Pet. 7 (citing Ex. 1012 ¶ 7.3.2).

Relying on the testimony of Dr. Malek, Patent Owner contends a person of ordinary skill in the art would have been a person with a Bachelor of Science degree in computer science or a related field, and at least two or three years of experience working in software development, distributed systems, database design, or a related technology. PO Resp. 11–12 (citing Ex. 2001 ¶ 42). Patent Owner also contends that working experience could be substituted for formal university training. PO Resp. 12.

The parties propose similar levels of ordinary skill in the art and do not directly challenge the other's proposal. We consider the level of ordinary skill in the art to be reflected by the prior art of record. *See Okajima v. Bourdeau*, 261 F.3d 1350, 1355 (Fed. Cir. 2001). The prior art references, like the Specification of the '600 patent, relate to object oriented and Java programming. *See* Ex. 1001, col. 20, l. 53–col. 38, l. 20 (including various example code excerpts); Ex. 1006, Abstract (describing invention as a Java

application); Ex. 1009, Title, xxvi (A programming guide for developing web applications in Java titled “Java for the Web with . . . EJB”).

In general, we agree with the areas of agreement in the parties’ proposals. Petitioner has not explained sufficiently why a graduate degree would be necessary, and so we adopt Patent Owner’s proposed level of a Bachelor of Science, which comports with the level of ordinary skill in the art reflected in the prior art of record. We determine that a person of ordinary skill would have possessed a Bachelor of Science degree in computer science (or related field, including electrical engineering) and at least two years of experience working in software development or database design (or a related technology, including distributed systems), and allowing additional commensurate work experience to substitute for formal academic training.

D. Obviousness over Demuth

Petitioner contends claims 1, 2, 6, and 8 are unpatentable under 35 U.S.C. § 103(a) as obvious over Demuth, relying on declaration testimony of Dr. Hosking. Pet. 19–38 (citing Ex. 1012). Patent Owner responds, relying on declaration testimony of Dr. Malek. PO Resp. 20–42 (citing Ex. 2001). Having considered the parties’ contentions and supporting evidence, we determine that Petitioner has demonstrated by a preponderance of evidence that claims 1, 2, 6, and 8 are unpatentable for obviousness over Demuth.

1. Summary of Demuth

Demuth describes techniques for minimizing the cost of developing object-oriented computer applications, for instance, by using software

“containers” as computer application building blocks. Ex. 1006, Abstract, ¶¶ 2, 4. Demuth’s techniques address the problem of developing object-oriented computer applications to run on a user’s computer. *Id.* at Abstract, ¶¶ 2, 19. Demuth’s solution takes advantage of computer logic available from the application components on the server computer—specifically, on the computer logic available from Enterprise JavaBeans (“EJB”) hosted computer applications. *Id.* at Abstract. To do so, Demuth describes a “client container” that is software that provides a web browser access to a subset of data and programming logic that otherwise is available from an EJB application. *Id.* at ¶¶ 2, 5, 8. The EJB application logic provides web pages for display on a user’s computer and provides a variety of programming infrastructure, including handling persistence. *Id.*

More particularly, Demuth’s “client container” software “transparently localizes” parts of a server-based computer application (i.e., an EJB application running on a server computer) so that the specified portions of the application can be accessed by the user’s computer (“client computer”) and “later re-persisted [from the client container] to the hosting EJB container.” *Id.* at Abstract; *see also id.* at ¶ 20. The “client container” software handles the “programming tasks associated with managing state locality for persistent objects [and] object access” and “hand-coding” is not required. *Id.* at ¶ 26.

Demuth also explains that the content and structure of the object-oriented computer application is composed, in part, of a “domain object model.” *Id.* at ¶ 56. Demuth notes that a domain object model is “familiar” to any person who develops object-oriented computer applications. *Id.* at ¶ 68. A domain object

model is made up of multiple “enterprise java beans” or “EJBs” in an “object graph.” *Id.* at ¶ 56. Demuth indicates a “domain object graph” is a giant web, or graph, with graph nodes representing objects and with graph edges (or lines connecting the nodes) as “navigable relations” between objects. *Id.*

2. Independent Claim 1

Independent claim 1 is a system claim that requires various types of information recited in elements (a)–(d); a computer-implemented method that includes an object (or set of objects) and computer logic for persisting an indicated object (or set of objects); an input method to provide the computer logic specified in the computer-implemented method with the location of recited information; and a data source in which persisted data may be stored.

In general, Petitioner asserts that Demuth’s transparent localization of EJB container logic in the “client container” software teaches or suggests the elements of claim 1. Pet. 29–38; *see also id.* at 19–29. The parties do not dispute that Demuth discloses lightweight representations of an object model and lightweight objects are transported from a server to a client, and that “changes to these lightweight objects are later reconciled back to a domain object graph on the EJB server.” PO Resp. 20; *see, e.g.*, Ex. 1006 ¶ 22 (Demuth’s description of the “client container” software). Patent Owner contends that Demuth does not teach a system for “creating or maintaining transparent persistence” (PO Resp. 20–21, 30–31, 33–35) and “is directed towards a server to client relationship” with reconciliation to a server rather than persistence to a database (*id.* at 21).

For the reasons explained in more detail below, we are persuaded that Demuth’s “client container” software that provides localization of lightweight objects and object models with persistence of changed objects back to the server teaches or suggests the subject matter of claim 1 as a whole. This is pertinent question under 35 U.S.C. § 103(a)—whether the claimed subject matter as a whole would have been obvious to one of ordinary skill in the art in view of Demuth, not merely whether Demuth discloses the subject matter of each element of claim 1 individually. *See In re Keller*, 642 F.2d 413, 425 (CCPA 1981) (“the test [for obviousness] is what the combined teachings of the references would have suggested to those of ordinary skill in the art”).

Patent Owner’s contentions regarding Demuth unduly focus on specific, isolated capabilities described in Demuth without addressing what those capabilities would have suggested to one of ordinary skill in the art at the time of the invention of the ’600 patent. Further, Patent Owner’s contentions in large measure amount to attacks on the individual elements of claim 1, without sufficient consideration of what the disclosure of Demuth would have suggested to one of ordinary skill in the art regarding the claimed subject matter as a whole, which is an approach we find unpersuasive. Indeed, many of Patent Owner’s contentions are more appropriate to a ground of anticipation, which requires a prior art reference to disclose, expressly or inherently, every limitation of the claim as arranged in the claim, *Net MoneyIN, Inc. v. VeriSign, Inc.*, 545 F.3d 1359, 1369 (Fed. Cir. 2001), rather than a ground of obviousness, which requires an analysis of whether the differences between the claimed subject matter and the prior art are such that the subject matter, as a whole,

would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains, 35 U.S.C.

§ 103(a). Notably, we instituted on Petitioner’s asserted ground of obviousness over Demuth, not on Petitioner’s asserted ground of anticipation by Demuth. Inst. Dec. 9 (identifying anticipation by Demuth as an asserted ground), 26 (Order instituting *inter partes* review for claims 1, 2, 6, and 8 based on obviousness over Demuth). Further, the challenged claims, when properly construed, do not require creating or maintaining transparent persistence, for the reasons explained in § II.A.1 or a database for the reasons explained below.

Accordingly, we find that Petitioner has shown by a preponderance of evidence that Demuth teaches or suggests all the claim elements and has articulated sufficient reasoning for the conclusion of obviousness. For the reasons explained below, we are not persuaded by Patent Owner’s arguments to the contrary. *See* PO Resp. 20–39.

Element (a)—a set of definitions for the relationships between a data source schema and objects capable of storing data for an object language application, wherein the set of definitions is stored in a repository

Demuth teaches or suggests element (a)—“a set of definitions for the relationships between a data source schema and objects capable of storing data for an object language application, wherein the set of definitions is stored in a repository.” Pet. 21–22 (Petitioner asserting the same). Demuth describes an EJB container that includes application logic and data access logic. Ex. 1006 ¶ 47. In particular, Demuth describes “data objects responsible for accessing data and presenting a view of the data,” which are called “entity beans.” *Id.* at

¶ 48. “The EJB container loads the appropriate entity beans [and] maps the entity beans to the database” using various enumerated techniques. *Id.* at ¶ 51.

Demuth’s mapping of an entity bean (which is an object responsible for accessing data and presenting a view of the data) to a database discloses “a set of definitions for the relationships between a data source schema and objects capable of storing data for an object language application,” as recited in claim 1. *Id.* We also credit the testimony of Petitioner’s declarant, Dr. Hosking, that the mapping of an entity bean to a database in Demuth discloses a set of definitions for the relationships between a data source schema and objects capable of storing data. *Id.* (citing Ex. 1012 ¶ 10.2.2.2.2).

Being mindful of the constructions of “a data source schema” as “an abstraction describing data in a data base used as a source of data in an application” and the construction of “object” as “a logic unit in a computer application or a computer software program based on an object-oriented programming language (e.g., Java) in which, depending on context, the logic unit refers either to an instance or to a class as a template,” we determine Demuth’s mapping of an entity bean to a database teaches or suggests “a set of definitions for the relationships between a data source schema and objects capable of storing data for an object language application,” as recited in claim 1.

Relying on Dr. Hosking’s testimony, Petitioner also contends that storing Demuth’s definitions in a repository, as required by claim 1, would have been obvious to one of ordinary skill in the art, who would have done so to enable the relationships to be retrieved later. Pet. 29 (citing Ex. 1012 ¶ 10.2.3.3.2).

Patent Owner, challenging Petitioner's position, contends that Demuth's set of definitions for the relationships between a data source schema and objects are stored within the EJBs or objects, rather than being stored within a repository, and contends storing the relationships in a repository would not have been obvious. PO Resp. 22–23 (citing Ex. 2001 ¶ 90). Patent Owner's declarant Dr. Malek testifies that “[n]othing in Demuth suggests departing from storing the connection information in the EJBs.” Ex. 2001 ¶ 90.

We credit Dr. Hosking's testimony over that of Dr. Malek, because Dr. Malek does not address persuasively Dr. Hosking's testimony that storing Demuth's definitions in a repository to enable the relationships to be retrieved later would have been obvious. It is within our discretion to assign the appropriate weight to the testimony offered by Dr. Malek and Dr. Hosking. *See, e.g., Yorkey*, 601 F.3d at 1284; *Am. Acad. of Sci. Tech Ctr.*, 367 F.3d at 1368.

Moreover, Demuth plainly teaches software (i.e., the EJB container) that maps entity beans to a database, which discloses the recited relationships to be stored in a repository. Patent Owner has not provided persuasive evidence how a person of ordinary skill in the art would understand the term “repository” to be a particular type of storage that would preclude storage of the relationships in Demuth's software container, as Demuth discloses. Further, storing the relationships in a repository would predictably use known elements (i.e., a repository) according to established functions (i.e., to store data), rendering a predictable result of storing the relationships in a repository, rather than in Demuth's software container. *See KSR*, 550 U.S. at 416 (“The combination of

familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results.”); Ex. 1017 ¶ 6.3.3 (Dr. Hosking testifying that storing definitions in a repository applies a known technique to yield predictable results).

Although Patent Owner contends Demuth does not teach element (a), Patent Owner does not address adequately the question of what the storage of Demuth’s relationships in a software container would have suggested to one of ordinary skill in the art. *Cf. Keller*, 642 F.2d at 425. For these reasons, we determine that Demuth teaches or suggests element (a).

Element (b)—a set of definitions for the relationships between objects for an object language application, wherein the set of definitions is part of an object application navigation model; Element (c)—a list of objects, or a set of objects that are to be persisted, wherein the list of objects or set of objects is part of an object application navigation model

Turning to elements (b) and (c) recited in claim 1, according to Petitioner with support from Dr. Hosking, the relationships between objects in Demuth’s domain object graph also teach the recited object application navigation models. Pet. 22–23 (citing Ex. 1012 ¶¶ 10.2.2.3.2, 10.2.2.4.2). More particularly, according to Petitioner and Dr. Hosking, Demuth’s subset of a domain object graph discloses “a set of objects that are to be persisted, wherein the . . . set of objects is part of an object application navigation model.” Pet. 23 (citing Ex. 1006 ¶ 70). As noted previously, Demuth’s domain object graph includes edges which represent relationships between objects, and Demuth’s edges are “navigable.” Ex. 1006 ¶ 68. Demuth further provides an example of “a subset of the domain object graph” that includes all objects needed for a user to

perform a function of the computer application, such as a user changing the tax withholding for an employee in a Human Resources system. Ex. 1006 ¶ 70. This teaches an example of Demuth's domain object graph being navigated for a certain purpose—changing the tax withholding for an employee.

Further, as Petitioner recognizes, Demuth teaches a set of objects that are to be persisted as part of an object application navigation model, as required by element (c). Pet. 23, Reply 10–12. First, Demuth discloses that the set of objects themselves can be sent back to the server as part of persisting the EJB container on the server. Reply 11 (citing Ex. 1017 ¶ 6.5). Demuth's lightweight object graph is a subset of Demuth's domain object graph, which teaches the recited object application navigation model, for the reasons noted previously. Thus, we determine that Demuth teaches or suggests a set of objects to be persisted (i.e., Demuth's lightweight objects) that is part of the object application navigation model (i.e., Demuth's domain object graph), as required by element (c).

Regarding elements (b) and (c), Patent Owner contends that Demuth does not teach the recited object application navigation model for several reasons. PO Resp. 23–30. First, Patent Owner contends that Demuth's domain object graph is a graph of the data objects that hold application data, and so does not include a description of expected behaviors when object relationships are navigated. *Id.* at 24–26. Therefore, according to Patent Owner, Demuth's domain object graph does not teach or suggest the recited object application navigation model. *Id.* Rather, Patent Owner contends that Demuth's domain object graph corresponds to the complex data object graph (CDOG) described

in the Specification of the '600 patent and so is not distinct from the CDOG as required in order for Demuth's domain object graph to render obvious the claimed object application navigation model. *Id.* at 26–27. The fact that Demuth's domain object graph also may teach the complex data object graph described by the '600 patent is not germane to whether Demuth's domain object graph teaches the recited object application navigation model. For the reasons explained previously in § II.A.4, the recited object application navigation model need not necessarily be separate from a data object graph and so it does not preclude Demuth's domain object graph from teaching both the recited object application navigation model and the complex data object described in the Specification of the '600 patent.

Patent Owner also contends that Demuth does not disclose a list of objects or a set of objects that is part of an object application navigation model because Demuth's subset of a data object graph includes all of the data objects needed to perform a particular function and is not limited to only changed objects, as purportedly required by claim 1. PO Resp. 29. Patent Owner acknowledges that Demuth sends back to the server for synchronization only the objects changed on the client. *Id.* (citing Ex. 1006 ¶ 90). Demuth's changed objects are part of the subset of a data object graph, which teaches or suggests objects being a part of an object application navigation model. Demuth's changed objects are sent to the server to be synchronized, which teaches or suggests the changed objects are a set of objects to be persisted. Accordingly, Demuth's description that changed objects sent to the server for

synchronization teaches or suggests “a set of objects that are to be persisted [and] the set of objects is part of an object application navigation model.”

For these reasons, we determine Demuth teaches or suggests elements (b) and (c) recited in claim 1.

Element (d)—an object or set of objects as part of a computer implemented method that contains the computer logic capable of persisting an indicated object or set of objects

Claim 1 also recites “an object or set of objects as part of a computer implemented method that contains the computer logic capable of persisting an indicated object or set of objects,” as element (d). As such, claim 1 recites two disjunctive phrases—(i) an object or set of objects and (ii) persisting an indicated object or set of objects. Notably, element (d) does not use antecedent basis to connect any of the objects (or sets of objects) or the computer implemented method to any other element recited in claim 1.

Demuth teaches or suggests element (d) recited in claim 1. Petitioner contends Demuth’s “client container” software that “transparently localizes” aspects of the object-oriented application on the server that are “later re-persisted” to the server teaches or suggests the “object or set of objects,” recited in element (d). Ex. 1006, Abstract (describing “client container” software); Pet. 23–24; Reply 12–13. Petitioner, with support from Dr. Hosking, contends Demuth’s “client container” teaches or suggests the recited persisting computer logic. Pet. 23.

Demuth’s “client container” software includes “lightweight object models . . . created as a subset of the main domain model construed as EJBs on

the server.” Ex. 1006 ¶ 22. Figure 5 of Demuth shows a client container 512 and is set forth below:

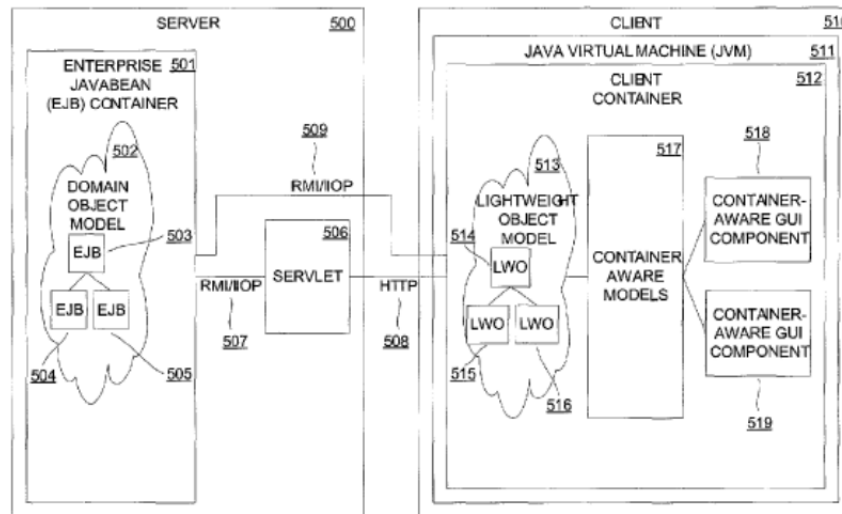


FIG. 5

As shown in Figure 5, Demuth’s client container 512 includes lightweight object model 513 having lightweight objects 514, 515, 516. *Id.* at ¶¶ 56–57. “The client container [software] provides lifecycle management of the lightweight object model by providing . . . delta tracking” of changes. *Id.* at ¶ 22. After the lightweight object model is modified by the user, the “client container” software “resolves back” the modified lightweight object model “into the heavyweight object model on the server.” *Id.*

Thus, Demuth describes “client container” software that includes computer logic capable of persisting a lightweight object model to the corresponding object model on the server. Accordingly, we agree with Petitioner that Demuth teaches or suggests an object or objects (i.e., lightweight object models) as part of a computer implemented method (i.e., client

container) that contains computer logic capable of persisting an indicated object or set of objects (i.e., lightweight objects). *See, e.g., id.* at Fig. 5.

Patent Owner contends that Demuth’s “client container” software does not disclose the recited persisting computer logic. PO Resp. 31–33. According to Patent Owner, Demuth’s “client container” software is not an object. *Id.* at 31. Patent Owner interprets element (d) as requiring an object (or set of objects) to contain the computer logic. Patent Owner’s narrow reading ignores “as part of a computer method” that immediately precedes “that contains the computer logic.” Dr. Malek’s testimony indicates an object, in object-oriented programming, “provides encapsulation of data and operations possible on the data.” Ex. 2001 ¶ 23. Thus, Dr. Malek’s testimony further supports the interpretation that element (d) requires the recited “computer implemented method” to contain the recited computer logic, which describes operations to be performed by the method. Thus, element (d) requires the recited “computer implemented method” to contain computer logic.

Moreover, even if element (d) requires the “object or set of objects” that are part of the computer-implemented method to contain the computer logic, based on Dr. Hosking’s testimony (Ex. 1017 ¶ 6.6.2), we find that the “client container” software containing computer logic at least suggests an object because the “client container” is a programming construct in an object-oriented programming paradigm. *See* Reply 13 (citing Ex. 1017 (Rebuttal Declaration of Dr. Hosking) ¶ 6.6).

Nor are we persuaded by Patent Owner’s contention that the computer logic in Demuth’s “client container” software, which expressly is described as

re-persisting objects to the server, is not the persisting computer logic required by element (d) because, according to Patent Owner, the “client container” software “only persists objects back to the EJB container on the server” and not to the database, as required by the claim. PO Resp. 33. As acknowledged by Patent Owner, persisted data may be stored to a database. *See, e.g.*, PO Resp. 36 (stating “one of ordinary skill in the art would understand that the persisting computer logic refers to computer logic that stores the changed state values back to the primary storage, *i.e.*, the database”).

Further, Patent Owner’s arguments unduly focus on specific, isolated capabilities described in Demuth without addressing what those capabilities would have suggested to one of ordinary skill in the art at the time of the invention of the ’600 patent. 35 U.S.C. § 103(a) (indicating the obviousness inquiry is whether the claimed subject matter as a whole would have been obvious to one of ordinary skill in the art in view of the applied prior art); *cf. Keller*, 642 F.2d at 425.

For these reasons, we determine Demuth teaches or suggests element (d).

Element (e)—an input method to provide the computer logic of d) with data corresponding to the location of information related to a), b) and c)

Claim 1 recites “an input method to provide the computer logic of d) with data corresponding to the location of information related to a), b) and c)” as element (e). Petitioner acknowledges that Demuth does not disclose expressly the recited input method. Petitioner contends, however, relying on Dr. Hosking’s testimony, that one of ordinary skill in the art would understand that an input method would be needed to allow a user to modify Demuth’s

objects or relationships. Pet. 30 (citing Ex. 1012 ¶ 10.2.3.7.2-5). We find Dr. Hosking’s testimony credible. It is within our discretion to assign the appropriate weight to the offered testimony. *See, e.g., Yorkey*, 601 F.3d at 1284.

Patent Owner contends that Demuth does not disclose element (e) because Demuth does not disclose the computer logic of element (d), which is required by element (e). For the reasons described previously, we determine that Demuth teaches or suggests computer logic of element (d) and so are not persuaded by Patent Owner’s contentions.

Accordingly, we determine Demuth teaches or suggests element (e).

Element (f)—at least one data source to in which persisted data may be stored

Claim 1 recites “at least one data source to in which persisted data may be stored” as element (f). Petitioner contends Demuth teaches or suggests element (f) because Demuth discloses the server’s EJB container may access a database and provides persistence logic. Pet. 25 (citing Ex. 1009 ¶¶ 41, 51). As acknowledged by Patent Owner, persisted data may be stored to a database. *See, e.g., PO Resp. 36* (stating “one of ordinary skill in the art would understand that the persisting computer logic refers to computer logic that stores the changed state values back to the primary storage, *i.e.*, the database”).

Notably, element (f) does not use antecedent basis to require any of its features to be related to any other element recited by claim 1. For instance, in Demuth, the data source need not necessarily be a database for storing data changed by the “client container” software.

For the foregoing reasons, we determine Demuth’s description of a server container that accesses a database and provides persistence logic teaches or suggests element (f).

3. Dependent Claims 2 and 8

Claims 2 and 8 each depend from independent claim 1 and each recite the system “that does not require any modifications to an object model or the inclusion of any persistence byte code in the object model in order to provide persistence for all or a portion of the complex data object graph model.”

Claim 8 additionally recites “a generic ejb stateful session bean.” Initially, we note the broad scope of this additional limitation, because it recites “a generic ejb stateful session bean” that lacks any connection to any other feature recited by claims 1 or 8.

For these additional limitations, Petitioner relies on Demuth’s description that “[a]ll of the infrastructure programming tasks associated with managing state locality for persistent objects, decoupling of state and behavior, object access and so on, are managed by the [client] container and do not require hand-coding.” Pet. 26–27. Petitioner also relies on Demuth’s description that the EJB container is responsible for managing state locality for persistent objects and Demuth’s indication that one type of EJB is a session bean. Pet. 27 (citing Ex. 1006 ¶¶ 26, 48, 51).

Demuth’s client container software expressly manages, without hand-coding, infrastructure tasks associated with managing state locality for persistent objects teaches or suggests providing persistence without requiring modifications to an object model or including of byte code in the object model,

which would require hand coding. As such, Demuth teaches or suggests the transparent persistence required by claims 2 and 8. See § II.A (construing transparent persistence and byte code).

Regarding the additional limitation recited by claim 8, based on Demuth's description that one type of EJB container is an EJB session bean (Ex. 1006 ¶ 48), we determine that Demuth teaches or suggests the EJB session bean, recited in claim 8.

Patent Owner contends that Demuth does not teach transparent persistence, relying on Dr. Malek's testimony. PO Resp. 39 (citing Ex. 2001 ¶ 140). Rather, according to Patent Owner, Demuth concerns "a server to client relationship, without disclosing a means for persisting to a database" and describes perceived shortcomings of the EJB container. *Id.* at 39–40.

We are not persuaded because Demuth's *client* container software, not the EJB container on the server, teaches or suggests transparent persistence for the reasons discussed previously. Further, Patent Owner does not address what these disclosures in Demuth would have suggested to one of ordinary skill in the art.

For the foregoing reasons, we determine that Demuth teaches or suggests the additional limitations recited in claims 2 and 8.

4. Claim 6

Claim 6, which depends from independent claim 1, additionally recites displaying, updating or creating the data or structure of one or more data objects, and optionally its links to other objects of an object model, by utilizing a displayable presentation page format wherein the displayable presentation page source code contains

embedded object programming logic, and persisting one or more of a member selected from the group consisting of an object, the data of an object, links of an object, or an object model by having the object programming logic that is embedded in the displayable presentation page directly or indirectly communicate with at least one object having at least one computer implemented method that includes imbedded computer logic for getting, setting, resetting and loading of data, and wherein the at least one object also includes computer implemented logic for communicating directly or indirectly with a persistence manager API or persistence library to persist one or more members selected from the group consisting of an object, the data of an object, a link to another object in an object model, and an object model.

For the reasons noted in § A.II.6 (construing “optionally” as indicating features not necessary to the claimed system), the limitation “optionally it[] links to other objects of an object model” is not required. Claim 6, among other limitations, includes (i) displayable presentation page source code containing embedded object programming logic that communicates with another object with enumerated computer logic and (ii) an object including logic for communicating with a persistence manager application programming interface (“API”) or a persistence library to perform enumerated functions.

Petitioner relies on Demuth’s description of how a user request for a web page is satisfied using Java Server Pages. Pet. 31–33 (citing Ex. 1006 ¶¶ 43, 50). Demuth describes a process, using JSP, to access data and computer logic. Ex. 1006 ¶ 50. When a user requests data by selecting a hyperlink on a web page generated by JSP, the JSP calls computer logic (“a session bean method”) that uses logic in other session beans. For example, the logic of a “session bean[] may locate the home interface for [particular] entity beans in which data

access methods are described, [and] then uses the home interface to ‘look up’ the specific entity bean[] instances representing required data.” For instance, Demuth teaches or suggests using a web page with a hyperlink (i.e., the presentation page format) to look-up data access methods and instances representing the required data (i.e., data or structure of data object). Thus, Demuth teaches or suggests the recited “displaying, updating or creating the data or structure of one or more data objects . . . by utilizing a displayable presentation page format” as required by claim 6.

Petitioner also relies on Demuth’s description of how a graph of lightweight objects (or a subset of a domain object graph) is retrieved. Pet. 31–33 (citing Ex. 1006 ¶ 119). This teaches or suggests an object communicating directly or indirectly an API or library. Petitioner further relies on Dr. Hosking’s testimony that it would have been obvious to use an API or library to persist an object, the data of an object, a link to another object in an object model, or an object model. Pet. 33.

Based on the foregoing reasons, we determine that Demuth teaches or suggests the additional limitations recited in claim 6.

5. Secondary Considerations

Although we have determined Demuth teaches or suggests the subject matter recited in claims 1, 2, 6, and 8, our inquiry continues because evidence of secondary considerations, when present, must always be considered as part of an obviousness inquiry. *Transocean Offshore Deepwater Drilling, Inc. v. Maersk Drilling USA, Inc.*, 699 F.3d 1340, 1349 (Fed. Cir. 2012).

Factual inquiries for an obviousness determination include secondary considerations based on evaluation and crediting of objective evidence of nonobviousness. *Graham v. John Deere Co.*, 383 U.S. 1, 17 (1966). Notwithstanding what the teachings of the prior art would have suggested to one with ordinary skill in the art at the time of the '600 patent's invention, the totality of the evidence submitted, including objective evidence of nonobviousness, may lead to a conclusion that the challenged claims would not have been obvious to one with ordinary skill in the art. *In re Piasecki*, 745 F.2d 1468, 1471–72 (Fed. Cir. 1984). Secondary considerations may include any of the following: long-felt but unsolved need, failure of others, unexpected results, commercial success, copying, licensing, and praise. *See Graham*, 383 U.S. at 17; *Leapfrog Enters., Inc. v. Fisher-Price, Inc.*, 485 F.3d 1157, 1162 (Fed. Cir. 2007).

To be relevant, evidence of nonobviousness must be reasonably commensurate in scope with the claimed invention. *In re Kao*, 639 F.3d 1057, 1068 (Fed. Cir. 2011) (citing *In re Tiffin*, 448 F.2d 791, 792 (CCPA 1971)); *In re Hiniker Co.*, 150 F.3d 1362, 1369 (Fed. Cir. 1998). More fundamentally, to be accorded substantial weight, there must be a nexus between the merits of the claimed invention and the evidence of secondary considerations. *GPAC*, 57 F.3d at 1580. “Nexus” is a legally and factually sufficient connection between the objective evidence and the claimed invention, such that the objective evidence should be considered in determining nonobviousness. *Demaco Corp. v. F. Von Langsdorff Licensing Ltd.*, 851 F.2d 1387, 1392 (Fed. Cir. 1988). The

burden of showing that there is a nexus lies with the Patent Owner. *Id.*; see *Paulsen*, 30 F.3d at 1482.

Patent Owner alleges its CocoBase product embodies the features of claim 1⁹ and “has received widespread industry recognition and awards for its advancements relating to transparent persistence.” PO Resp. 41. In particular, Patent Owner identifies a WebSphere Advisor Editors’ Choice Award was awarded to CocoBase in 2003 “specifically [for] the transparent persistence present in claim 1.” PO Resp. 41–42 (citing Ex. 2004, 4). Patent Owner also indicates CocoBase in 2003 was named the “Best of the Net” by About.com in a particular category because “CocoBase does not require EJB, JDBC, or SQL coding.” PO Resp. 42 (citing Ex. 2005, 1).

Patent Owner, however, fails to establish a nexus between the merits of the claimed invention and the evidence of secondary considerations. Patent Owner’s broad contentions about “transparent persistence,” which is not required by claim 1, do not provide a sufficient connection between objective evidence and the claimed invention, and so do not establish the requisite nexus between the merits of the claimed invention and the evidence of secondary considerations. Moreover, as noted previously, the general concept of creating or maintaining transparent persistence was known at least as early as November 2001, sixteen months before the effective filing date of the application that

⁹ Although Patent Owner only contends that secondary considerations guide against a finding that claim 1 would have been obvious over Demuth, we consider Patent Owner’s objective evidence in view of the subject matter of all challenged claims.

issued as the '600 patent. *See* Ex. 1016, 2 (Sun Microsystem article describing CocoBase in November 2001 as providing “dynamic transparent persistence” to allow a software programmer “to persist relational data with a simple, non-invasive . . . object” and “[w]ithout any byte code or object model intrusion.”). *See Ormco Corp. v. Align Tech., Inc.*, 463 F.3d 1299, 1311–12 (Fed. Cir. 2006) (“[I]f the commercial success is due to an unclaimed feature of the device, the commercial success is irrelevant. So too if the feature that creates the commercial success was known in the prior art, the success is not pertinent.”); *J.T. Eaton & Co. v. Atl. Paste & Glue Co.*, 106 F.3d 1563, 1571 (Fed. Cir. 1997) (“asserted commercial success of the product must be due to the merits of the claimed invention beyond what was readily available in the prior art”). Claims 2 and 8 each recite a system that “does not require any modifications to an object model or the inclusion of any persistence byte code in the object model in order to provide persistence for all or a portion of the complex data object graph model.” These limitations are features of transparent persistence. The objective evidence provided by Patent Owner, however, does not provide a sufficient connection between objective evidence and the claimed invention—the limitations recited in claims 2 and 8.

Patent Owner does not provide sufficient evidence that awards address the features recited by claims 2 and 8, and, therefore, does not established the required nexus. Accordingly, Patent Owner fails to provide sufficient credible evidence to support its allegations of nonobviousness based on secondary considerations. Balancing Petitioner’s evidence of obviousness against Patent Owner’s asserted objective evidence of nonobviousness, we determine that a

preponderance of the evidence supports Petitioner's position that claims 1, 2, 6, and 8 would have been obvious over Demuth.

*6. Conclusion Regarding Obviousness of
Claims 1, 2, 6, and 8 in view of Demuth*

For the foregoing reasons, we determine that the subject matter recited in each of claims 1, 2, 6, and 8 as a whole would have been obvious to one of ordinary skill in the art in view of Demuth. *See* 35 U.S.C. § 103(a). Patent Owner's arguments in large measure amount to attacks on Demuth's disclosure of individual elements recited in the claims, without sufficient consideration of whether the subject matter as a whole would have been obvious in view of the disclosure of Demuth and what the disclosure of Demuth suggested to one of ordinary skill in the art at the time of the invention of the '600 patent. We have resolved the question of obviousness based on factual determinations of (1) the scope and content of Demuth; (2) differences between the subject matter of claims 1, 2, 6, and 8 and the disclosure of Demuth; (3) the level of ordinary skill in the art; and (4) objective evidence of nonobviousness. *Graham*, 383 U.S. at 17–18. Therefore, we determine that Petitioner has shown by a preponderance of the evidence that the subject matter of claims 1, 2, 6, and 8 of the '600 patent would have been obvious to a person of ordinary skill in the art in view of the teaching of Demuth.

E. Obviousness over Demuth and Kurniawan

Petitioner contends claims 1, 2, 6, and 8 are unpatentable under 35 U.S.C. § 103(a) as obvious over Demuth and Kurniawan, relying on

declaration testimony of Dr. Hosking. Pet. 50–54 (citing Ex. 1012). Patent Owner responds, relying on declaration testimony of Dr. Malek. PO Resp. 43–49 (citing Ex. 2001). Having considered the parties’ contentions and supporting evidence, we determine that Petitioner has demonstrated by a preponderance of evidence that claims 1, 2, 6, and 8 are unpatentable for obviousness over Demuth and Kurniawan.

In general, Petitioner relies on its arguments that Demuth teaches or suggests the subject matter of claims 1, 2, 6, and 8, as discussed previously. Similarly, Patent Owner relies on similar arguments that Demuth does not teach or suggest the subject matter of claims 1, 2, 6, and 8, which we do not find persuasive for the reasons given previously.

Accordingly, we focus our discussion below on the limitations that Petitioner relies on Kurniawan for additional support and Petitioner’s reasoning to support its legal conclusion of obviousness over Demuth and Kurniawan.

1. Claims 6 and 8 in View of Demuth and Kurniawan

To further support its contention that claims 6 and 8 would have been obvious, Petitioner augments its contentions that the challenged claims would have been obvious over Demuth with assertions based on Kurniawan’s description of JavaServer Page (“JSP”) technology and “a stateful session bean.” Pet. 51–54. Kurniawan’s book is a guide to Java web programming that describes technologies for programming web applications in Java, including JSP, EJB, and programming on both servers and client computers. *See* Ex. 1009, Title; Pet. 50.

More particularly, Petitioner additionally relies on Kurniawan's description of JSP technology as disclosing the displayable presentation page, as recited in claim 6, which depends from independent claim 1. For example, Petitioner relies on Kurniawan's description of a Java Server Page container "to generate Java code that loads the bean's class" as disclosing or suggesting an "object having at least one computer implemented method that includes [e]mbedded computer logic for . . . loading of data." Pet. 52 (citing Ex. 1009, 306¹⁰). We agree with Petitioner and note that Patent Owner does not contest such a determination (PO Resp. 49).

Turning to claim 8, Petitioner relies on Kurniawan for the recited "a generic ejb stateful session bean." For this limitation, Petitioner relies on Kurniawan's description of a "stateful session bean":

[a] session bean is an enterprise bean that implements business logic for its clients. . . . The stateful session bean retains data related to the client that is retrieved from the database or information entered by the client. An instance of a client session is said to hold the *conversational state* of the client. A client session ends when the client specifies that it no longer needs the stateful session bean or after a certain period of time lapses since the last method invocation by the client.

Pet. 53-54 (citing Ex. 1009 at 687–88) (emphasis added).

We agree with Petitioner that Kurniawan discloses a stateful session bean as required in claim 8. Patent Owner does not contest this. PO Resp. 47–48.

¹⁰ Citations to Ex. 1009 refer to page numbers in the original book, not the exhibit page numbers added by Petitioner.

2. Reason for Combining Demuth and Kurniawan

“[T]here must be some articulated reasoning with some rational underpinning to support the legal conclusion of obviousness.” *KSR*, 550 U.S. at 418. Relying on Dr. Hosking’s testimony, Petitioner additionally contends that it would have been obvious to combine Demuth and Kurniawan for various reasons, including to improve a distributed processing system, to allow easier accessibility, and to provide uniformity and/or integrity for a system or application. *See* Pet. 53 (citing Ex. 1012 ¶¶ 10.2.6.12.3, 10.2.6.12.5); *see also id.* at 51–53 (citing Ex. 1012 ¶¶ 10.2.6.11.2, 10.2.6.11.3, 10.2.6.13.2).

Patent Owner asserts the Dr. Hosking merely provides a conclusory statement that the combination of Demuth and Kurniawan would have been obvious because the references are in the same field of endeavor. PO Resp. 43–44. Patent Owner, however, does not address sufficiently Dr. Hosking’s testimony regarding reasons one skilled in the art at the time of the invention would have combined Demuth and Kurniawan with regard to claim 6 or claim 8, as previously noted in part. *Compare* PO Resp. 43–44 *with* Pet. 51–52 (regarding claim 6); Pet. 53 (regarding claim 8).

In view of the foregoing, we are persuaded that Petitioner has articulated a sufficient reason to support a conclusion of obviousness in view of Petitioner’s combination of Demuth and Kurniawan, based on Dr. Hosking’s testimony, which is supported by sufficient articulated reasoning with rational underpinning.

3. Secondary Considerations

Patent Owner alleges its CocoBase product embodies the features of claim 1 and so is not obvious over Demuth (PO Resp. 41–42), without specifically asserting claim 1 would not have been obvious over Demuth and Kurniawan. Because secondary considerations, if present, should always be considered, we determine that, for the reasons discussed above, Patent Owner fails to provide sufficient credible evidence to support a finding of nonobviousness, based on secondary considerations, over Demuth and Kurniawan.

4. Conclusion Regarding Obviousness of Claims 1, 2, 6, and 8 in View of Demuth and Kurniawan

Accordingly, we determine that the subject matter recited in each of claims 1, 2, 6, and 8 as a whole would have been obvious to one of ordinary skill in the art in view of Petitioner's combination Demuth and Kurniawan. 35 U.S.C. § 103(a). Patent Owner's arguments fail to provide sufficient consideration of whether the subject matter in each of the claims as a whole would have been obvious in view of the disclosure and suggestions of the combination of Demuth and Kurniawan.

We have resolved the question of obviousness based on factual determinations of (1) the scope and content of Demuth and Kurniawan; (2) differences between the subject matter of claims 1, 2, 6, and 8 and the teachings of Demuth and Kurniawan; (3) the level of ordinary skill in the art; and (4) objective evidence of nonobviousness. *Graham*, 383 U.S. at 17–18. Therefore, we determine that Petitioner has shown by a preponderance of the

evidence that the subject matter of claims 1, 2, 6, and 8 of the '600 patent would have been obvious to a person of ordinary skill in the art in view of the teachings of Demuth and Kurniawan.

III. CONCLUSION

Petitioner has proven by a preponderance of the evidence that claims 1, 2, 6, and 8 of the '600 patent are unpatentable under 35 U.S.C. § 103(a) as obvious over Demuth and are unpatentable as obvious over Demuth and Kurniawan.

IV. ORDER

Accordingly, it is hereby

ORDERED that, based on a preponderance of the evidence, claims 1, 2, 6, and 8 of U.S. Patent No. 7,103,600 B2 are unpatentable; and

FURTHER ORDERED that, because this is a Final Written Decision, the parties to the proceeding seeking judicial review of the decision must comply with the notice and service requirements of 37 C.F.R. § 90.2.

IPR2014-00118
Patent 7,103,600 B2

For PETITIONER:
Vaibhav P. Kadaba
Steven D. Moore
Kilpatrick Townsend & Stockton LLP
wkadaba@kilpatricktownsend.com
smoore@kilpatricktownsend.com

For PATENT OWNER:
J. Warren Lytle, Jr.
Brian K. Shelton
SUGHRUE MION PLLC
jlytle@sughrue.com
bshelton@sughrue.com