

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

CHERWELL SOFTWARE, LLC,
Petitioner,

v.

BMC SOFTWARE, INC.,
Patent Owner.

Case IPR2018-00978
Patent 8,832,652 B2

Before JUSTIN T. ARBES, JOHN A. HUDALLA, and
CHRISTOPHER L. OGDEN, *Administrative Patent Judges*.

ARBES, *Administrative Patent Judge*.

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

I. INTRODUCTION

Petitioner Cherwell Software, LLC filed a Petition (Paper 2, “Pet.”) requesting *inter partes* review of claims 1–20 of U.S. Patent No. 8,832,652 B2 (Ex. 1001, “the ’652 patent”) pursuant to 35 U.S.C. § 311(a). Patent Owner BMC Software, Inc. filed a Preliminary Response (Paper 8, “Prelim. Resp.”) pursuant to 35 U.S.C. § 313. Pursuant to 35 U.S.C. § 314(a), the Director may not authorize an *inter partes* review unless the information in the petition and preliminary response “shows that there is a reasonable likelihood that the petitioner would prevail with respect to at least 1 of the claims challenged in the petition.” For the reasons that follow, we have decided not to institute an *inter partes* review.

II. BACKGROUND

A. *The ’652 Patent*

The ’652 patent discloses “a technique for customizing software applications.” Ex. 1001, col. 1, ll. 15–17. The ’652 patent describes a known problem in the art where a customer acquires a software package from a supplier and subsequently makes changes to the “logic, structures, data definition or capability” of the software to meet the customer’s needs. *Id.* at col. 1, ll. 18–27. When the supplier provides a patch or new release of the software, the customer may want to “apply [the] update to get the fixes and additional functionality of the new version from the supplier but still preserve the customer’s customizations,” which was difficult because it would typically require “redo[ing] the customizations on the new release.” *Id.* at col. 1, ll. 28–38. Another known problem arises when multiple groups used and customized the software package in their own ways, such that

multiple “independent implementations” are needed for each group. *Id.* at col. 1, ll. 39–48. The ’652 patent attempts to solve these problems by “overlaying objects of a software application with other objects [to] allow modification and customization of the application by one or more users in different ways” and “configuring the software application to execute using overlaid objects instead of the base objects contained in the software application.” *Id.*, Abstract, col. 3, ll. 55–60.

Figure 1 of the ’652 patent is reproduced below.

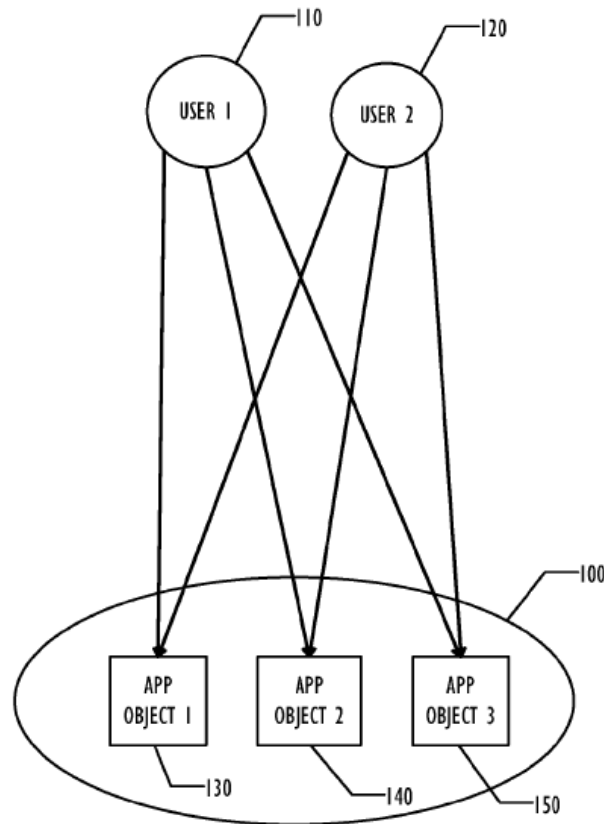


Figure 1 depicts a prior art arrangement where application 100, which has three application objects 130, 140, 150, may be executed by users 110, 120. *Id.* at col. 4, ll. 52–55. In the prior art arrangement, different copies of application 100 would be created and customized separately by the different users, making management difficult and requiring that all permutations be

re-created whenever the supplier updates application 100. *Id.* at col. 4, ll. 55–65.

Figure 2 of the '652 patent is reproduced below.

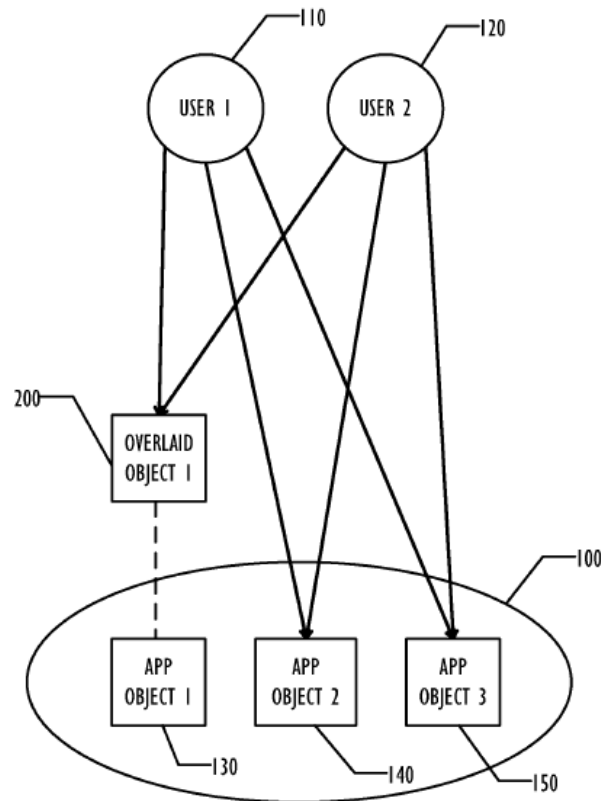


Figure 2 depicts the disclosed arrangement using overlaid object 200. “Both users 110 and 120 may use application 100 with overlaid object 200 overlaying or replacing original application object 130 when executing application 100.” *Id.* at col. 5, ll. 23–26. Application 100 can then be “modified by the vendor without affecting overlaid object 200, and users 110 and 120 may continue to use customized application 100 with overlaid object 200 even after an update by the vendor to application 100 modifies the application object 130.” *Id.* at col. 5, ll. 26–30. According to the '652 patent,

[b]y using overlays for customizations, the process of upgrading or patching the application when getting a new or

updated definition from the application provider or vendor may be dramatically simplified. The new definitions may simply be installed since all customizations are stored separately from the original source definitions. The customer need not be concerned with overwriting a customization or of making the requirement that customizations have to be made again. At the same time, the customizations may still [be] tied to the original definitions, so that the customized changes will still be in place, in the right location, and performing the same function as before the upgrade of the application from the application provider or vendor.

Id. at col. 5, l. 60–col. 6, l. 5.

The '652 patent discloses that “[a]ny construct of an application may be overlaid to have its functionality altered.” *Id.* at col. 5, ll. 41–45. Overlaid objects may change, for example, “characteristics of fields or data” (e.g., “the size of a field in a form or in a database table”) or “callable functions of a program.” *Id.* at col. 8, l. 58–col. 9, l. 4. In an interpretation environment, overlaying may be performed prior to execution of the program or at runtime, where the program decides to execute either a base function or overlaid function at each invocation of the function. *Id.* at col. 9, ll. 4–15, Fig. 8. In a compiled language environment, “a single executable may be generated that decides at runtime which overlaid object should be executed.” *Id.* at col. 9, ll. 16–28. Finally, the '652 patent discloses organizing “multiple, independent sets of overlay definitions” in “overlay group[s]” for different groups of users, with the groups being arranged in stacked levels of a hierarchy (Figure 4) or in parallel (Figure 5), and controlling access to overlaid objects via “permission groups” (Figure 6). *Id.* at col. 6, l. 18–col. 8, l. 35.

B. Illustrative Claim

Claim 1 of the '652 patent recites:

1. A method, comprising:

while a processor associated with a computer system is one of interpreting, in an interpreted computer environment, and compiling, in a compiled computer environment, a software application, determining at least one base object of the software application has at least one corresponding overlaid object;

associating, by the processor, a first overlaid object with a first base object of the software application, the first overlaid object being configured to change a construct definition of the first base object of the software application;

configuring, by the processor, the software application to execute in the computer system using the first overlaid object instead of the first base object; and

configuring, by the processor, the software application to execute in the computer system using the first base object instead of the first overlaid object.

C. The Prior Art

Petitioner relies on U.S. Patent No. 7,734,999 B2, which was filed on January 3, 2005, and issued on June 8, 2010 (Ex. 1003, “Leung”).

D. The Asserted Ground

Petitioner challenges claims 1–20 of the '652 patent as unpatentable over Leung under 35 U.S.C. § 103(a).¹

¹ The Leahy-Smith America Invents Act, Pub. L. No. 112-29, 125 Stat. 284 (2011) (“AIA”), amended 35 U.S.C. § 103. Because the challenged claims of the '652 patent have an effective filing date before the effective date of the applicable AIA amendment, we refer to the pre-AIA version of 35 U.S.C. § 103.

III. ANALYSIS

A. Claim Interpretation

The Board interprets claims in an unexpired patent using the “broadest reasonable construction in light of the specification of the patent in which [they] appear[.]” 37 C.F.R. § 42.100(b). Petitioner proposes interpretations for three claim terms: “object,” “overlaid object,” and “dictionary.”

Pet. 11–13. Patent Owner disagrees with Petitioner’s proposed interpretations, but states that claim interpretation is unnecessary because Petitioner fails to show that Leung teaches other limitations of the claims. Prelim. Resp. 6–15. We agree with Patent Owner as to certain of those other limitations pertaining to a “software application,” as explained herein, and thus conclude that no claim interpretation is necessary to determine whether to institute a trial in this proceeding. *See Nidec Motor Corp. v. Zhongshan Broad Ocean Motor Co. Ltd.*, 868 F.3d 1013, 1017 (Fed. Cir. 2017) (“Because we need only construe terms ‘that are in controversy, and only to the extent necessary to resolve the controversy,’ we need not construe [a particular claim limitation] where the construction is not ‘material to the . . . dispute.’” (citations omitted)).

B. Principles of Law

A claim is unpatentable for obviousness if, to one of ordinary skill in the pertinent art, “the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made.” *KSR Int’l Co. v. Teleflex Inc.*, 550 U.S. 398, 406 (2007) (quoting 35 U.S.C. § 103(a)). The question of obviousness is resolved on the basis of underlying factual

determinations, including “the scope and content of the prior art”; “differences between the prior art and the claims at issue”; and “the level of ordinary skill in the pertinent art.”² *Graham v. John Deere Co.*, 383 U.S. 1, 17–18 (1966).

A patent claim “is not proved obvious merely by demonstrating that each of its elements was, independently, known in the prior art.” *KSR*, 550 U.S. at 418. An obviousness determination requires finding “both ‘that a skilled artisan would have been motivated to combine the teachings of the prior art references to achieve the claimed invention, and that the skilled artisan would have had a reasonable expectation of success in doing so.’” *Intelligent Bio-Sys., Inc. v. Illumina Cambridge Ltd.*, 821 F.3d 1359, 1367–68 (Fed. Cir. 2016) (citation omitted); *see also KSR*, 550 U.S. at 418 (for an obviousness analysis, “it can be important to identify a reason that would have prompted a person of ordinary skill in the relevant field to combine the elements in the way the claimed new invention does”). A petitioner’s assertion of obviousness “cannot employ mere conclusory statements. The petitioner must instead articulate specific reasoning, based on evidence of record, to support the legal conclusion of obviousness.” *In re Magnum Oil Tools Int’l, Ltd.*, 829 F.3d 1364, 1380 (Fed. Cir. 2016) (citing *KSR*, 550 U.S. at 418).

² Additionally, secondary considerations, such as “commercial success, long felt but unsolved needs, failure of others, etc., might be utilized to give light to the circumstances surrounding the origin of the subject matter sought to be patented. As indicia of obviousness or nonobviousness, these inquiries may have relevancy.” *Graham*, 383 U.S. at 17–18. Patent Owner, however, has not presented any such evidence.

C. Level of Ordinary Skill in the Art

Section 103(a) forbids issuance of a patent when “the differences between the subject matter sought to be patented 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, 550 U.S. at 406 (quoting 35 U.S.C. § 103(a)). Petitioner argues that a person of ordinary skill in the art at the time of the ’652 patent would have had “a bachelor’s degree in computer science or computer engineering, or in a related field, and approximately two or three years of experience in industry with respect to software design and computer systems.” Pet. 10 (citing Ex. 1004 ¶ 33). Patent Owner does not address the level of ordinary skill in the art in its Preliminary Response. Based on the record presented, including our review of the ’652 patent and the types of problems and solutions described in the ’652 patent and cited prior art, we agree with Petitioner’s assessment of the level of ordinary skill in the art and apply it for purposes of this Decision.

D. Obviousness Ground Based on Leung

Petitioner contends that claims 1–20 are unpatentable over Leung under 35 U.S.C. § 103(a), citing the testimony of Jon B. Weissman, Ph.D., as support. Pet. 14–64 (citing Ex. 1004). Patent Owner makes various arguments in response. Prelim. Resp. 15–34. For the reasons explained below, we are not persuaded that Petitioner has established a reasonable likelihood of prevailing on its asserted ground.

1. Leung

Leung describes a system for “configuring a user formset for display on a user interface of a client computer.” Ex. 1003, Abstract. According to Leung, “forms implemented on a graphical user interface of a user computer [had] to be coded for operation by a variety of different users” by hand, such that “any changes in one [role-specific Formset] require[d] that all the other similar Formsets be examined and also changed if necessary.” *Id.* at col. 1, ll. 6–13, 20–25. Leung avoids that problem by applying overlays to a master formset to create a customized formset. *Id.* at col. 1, l. 25–col. 2, l. 15.

Figure 1 of Leung is reproduced below.

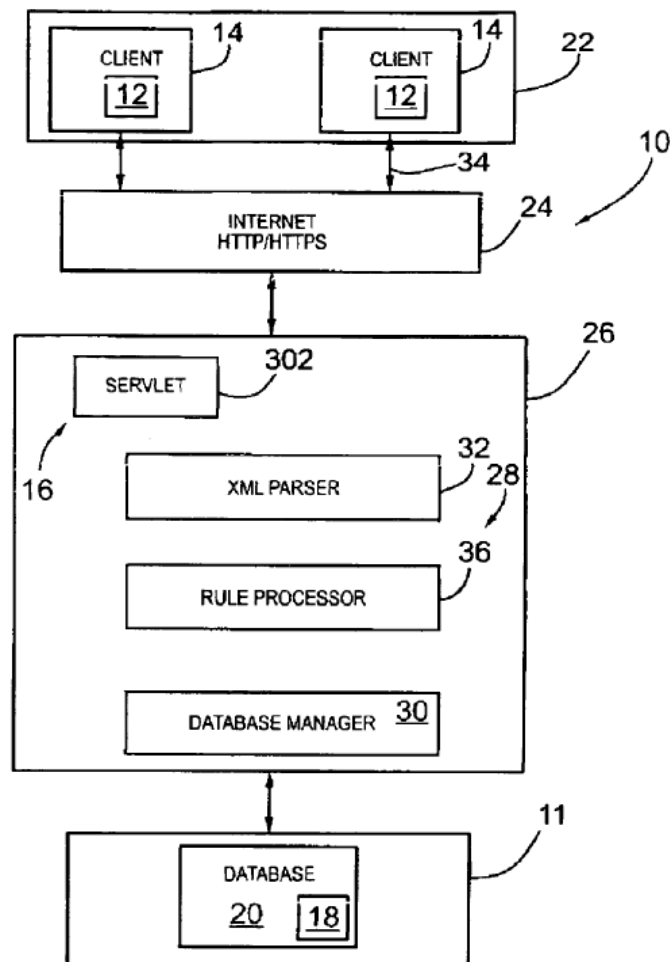


FIG. 1

Figure 1 depicts forms processing system 10 comprising client 14, which communicates over network 24 with form server 16 accessing database 20. *Id.* at col. 2, ll. 41–62. Form server 16 provides client formsets 12 (“i.e. a plurality of forms presented as multiple screens that are interconnected by defined workflow”) to client 14. *Id.* at col. 2, ll. 42–47. “[C]lient formsets 12 are dynamically configurable (i.e. customizable) based on user setting/type and priorities in relation to a master formset(s) 18 stored on a database 20 and/or the form server 16.” *Id.* at col. 2, ll. 50–53. “Each of the generated production formsets 12 are assigned to a role” (via “static overlays”) and “served up to the users of the client 14 in that role, such that the formset 12 look and functionality is changed based on the type/role of user, as well as individual settings” (via “dynamic overlays”). *Id.* at col. 2, l. 62–col. 3, l. 1.

Figure 3 of Leung is reproduced below.

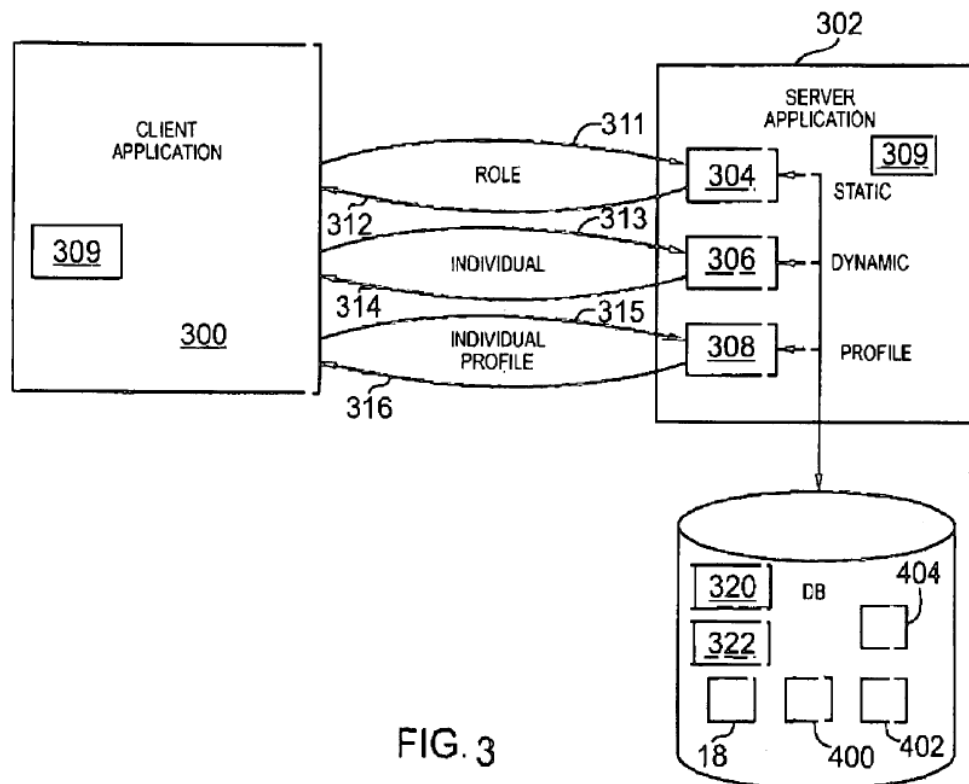


Figure 3 depicts how formsets are processed using client application 300, server application 302, and database 20. *Id.* at col. 4, l. 55–col. 5, l. 56.

“[C]lient application 300 can be a Java applet (or other application) that runs in connection with a browser” and “operates to dynamically render configurable client formsets 12” for display to the user. *Id.* at col. 4, ll. 56–63. “The workflow (e.g. actions and events) as well as layout of the formset[] 12 can be defined in [an Extensible Markup Language (XML)] file representing the formset 12,” which is transmitted from form server 16 to client 14. *Id.* at col. 4, ll. 63–66. Once client formset 12 is rendered using client application 300, the user enters or modifies data and submits the modified data to form server 16 as an XML transaction, after which server application 302 processes the transaction and returns another XML file for presentation to the user. *Id.* at col. 5, ll. 34–51.

Leung describes static and dynamic overlays that “modify the XML definitions in the master formset 18” to create a customized formset for the particular user. *Id.* at col. 5, ll. 30–33, Fig. 4. First, form server 16 identifies the user and his or her role (“e.g. administrator vs. a technician”), and server application 302 applies “static overlay/filter 400 to the master formset 18, such that the generic definitions are converted into role/category specific definitions” appropriate for all users of that role. *Id.* at col. 7, ll. 37–43, col. 8, l. 43–col. 9, l. 64. “[S]tatic overlays 400 can be either additive or restrictive in nature and can be used to add, modify, or delete objects and their related functionality available in the master formset 18.” *Id.* at col. 8, ll. 7–10. Second, server application 302 modifies the formset further by applying dynamic overlays 402 to customize for the individual user. *Id.* at col. 11, l. 46–col. 12, l. 37. Dynamic overlays 402 permit modifications in

what the individual user (or groups of users) see(s) and what they can change, based on an application specific way of grouping the users based on individual permissions and settings (e.g. Bob as a technician as compared to Charles) or as grouping the users based on grouped permissions and settings (e.g. technicians in department A as compared to technicians in department B). In the first case the technicians are the role/category while Bob and Charles are the individuals. In the second case the technicians are the category/role while the departments are the “individuals.”

Id. at col. 11, ll. 47–58. “The dynamic overlay 402 feature is specifically restrictive in nature and preferably does not add objects and their related functionality to the formset 12.” *Id.* at col. 8, ll. 33–35.

2. Claims 1–10

Petitioner argues that Leung teaches all limitations of the challenged claims, and that “to the extent they [are] not explicitly disclosed, they would have been obvious under 35 U.S.C. § 103 to a person of ordinary skill in the art.” Pet. 15. We focus on the limitations of claim 1 pertaining to a “base object” and “software application” that are disputed by Patent Owner. See Prelim. Resp. 22–27. In particular, claim 1 recites (1) “while a processor associated with a computer system” is interpreting or compiling “a software application, determining at least one base object of the software application has at least one corresponding overlaid object,” (2) “associating, by the processor, a first overlaid object with a first base object of the software application,” and (3) “configuring, by the processor, the software application to execute in the computer system.” Thus, claim 1 requires at least one base

object “of” a software application and that the software application be configured by a processor to “execute.”³

Petitioner has not provided sufficient explanation as to how Leung allegedly teaches these limitations. Petitioner includes in its Petition a general summary of what Leung discloses, Pet. 15–21, and a claim chart with underlined quotations from Leung, *id.* at 21–26. In those portions, Petitioner appears to argue that the generic XML definitions of master formset 18 constitute “base object[s]” as recited in claim 1. *See id.* at 15–16 (“The master formset’s base objects exist in the form of XML definitions.”), 19 (discussing “generic definitions” of master formset 18), 23 (underlining “the content of the master formset[] 18 (containing more generic definitions of content, presentation and workflow)”); Ex. 1004 ¶¶ 45, 57. Petitioner further appears to contend that the modified “role/category specific” XML definitions in client formset 12 (after application of static overlays 400 to master formset 18) constitute “overlaid object[s]” as recited in claim 1. *See* Pet. 17–18 (discussing “role-specific components”), 23–24 (underlining Leung’s disclosure that “generic definitions are converted into role/category specific definitions” and that static overlays 400 are used to “modify the XML definitions” in master formset 18); Ex. 1004 ¶¶ 45–46, 59–60.

It is unclear from the Petition, however, what specific component(s) in Leung Petitioner is relying on as allegedly teaching a “software application.”

³ An exemplary embodiment including these limitations is shown in Figure 2 of the ’652 patent above, which depicts application 100 “with 3 application objects 130, 140, and 150” that is “execute[d]” by users 110 and 120, and “object 130 of the application 100 . . . overlaid by overlaid object 200.” Ex. 1001, col. 4, ll. 52–55, col. 5, ll. 21–30.

Petitioner primarily refers to client application 300, but also mentions “an application (e.g., a formset)” and refers in various places in its claim chart to master formset 18 and client formset 12. *See* Pet. 15–26. A petition for *inter partes* review must include a “full statement of the reasons for the relief requested, including a detailed explanation of the significance of the evidence,” must identify “where each element of the claim is found in the prior art patents or printed publications relied upon,” and must “identify[] specific portions of the evidence that support the challenge.” 37 C.F.R. §§ 42.22(a)(2), 42.104(b)(4)–(5). “The Board may exclude or give no weight to the evidence where a party has failed to state its relevance or to identify specific portions of the evidence that support the challenge.” 37 C.F.R. § 42.104(b)(5). Petitioner’s general description of Leung and unexplained quotations from Leung in its claim chart fail to meet these requirements.

Regardless of whether Petitioner is relying on client application 300 or Leung’s formsets as a “software application,” though, we are not persuaded that Petitioner has established a reasonable likelihood of prevailing. With respect to client application 300, Petitioner argues that

Leung discloses that this application is executed by a “processor . . . by executing related instructions . . . provided by an operating system.” Leung discloses that the processor may interpret the application in an interpreted environment: “Referring to FIG. 3, the client application 300 can be a Java applet (or other application) that runs in connection with a browser presented on the UI 202 (see FIG. 2)” One of ordinary skill would understand Java to be an interpreted language.

Pet. 20–21 (citations omitted); *see also id.* at 21–22, 24 (quotations from Leung pertaining to client application 300); Ex. 1004 ¶ 52. As described in

the portions of Leung cited by Petitioner, client application 300 is executed by processor 208 of workstation computer 201, and may be a Java applet running in a browser. Ex. 1003, col. 4, ll. 16–24, 50–58, col. 5, ll. 15–19, Fig. 2. Client application 300, however, does not comprise any XML definitions itself—it merely processes the received XML file corresponding to client formset 12 and renders the forms on the client’s screen. *See id.* at col. 4, l. 55–col. 5, l. 19, col. 5, ll. 41–51, Fig. 2. Indeed, as Patent Owner correctly points out, Leung does not describe customizing client application 300 using overlays in any way. *See Prelim. Resp.* 24, 27. Rather, client application 300 is simply the vehicle by which the XML file containing the definitions is received. Thus, we are not persuaded by Petitioner’s arguments regarding client application 300 that Leung teaches at least one base object “of” a software application that is configured by a processor to “execute.”

To the extent Petitioner’s argument is that master formset 18 or client formset 12, rather than client application 300, is a “software application,” Petitioner does not explain in any detail how such a formset is configured, by the processor that performs the “associating” step, to “execute,” as recited in claim 1. *See* 37 C.F.R. §§ 42.104(b)(4)–(5). Again, Leung discloses formsets represented as XML files, where an XML file for client formset 12 is created by modifying master formset 18 and then transmitted to the client’s computer. Ex. 1003, col. 4, ll. 63–66, col. 5, ll. 30–33, col. 7, ll. 38–43. Once received, client application 300 (e.g., a Java applet running in a browser) processes the XML file and “operate[s] to dynamically render configurable client formsets 12 . . . on the browser.” *Id.* at col. 4, l. 56–col. 5, l. 22. Because Petitioner does not explain at all how the XML

file can be considered a “software application” that “execute[s],” how such actions constitute configuring the XML file to “execute,” or whether the XML file is configured to “execute” by the same processor and in the same computer system that performs the “associating” step, Petitioner has not made a sufficient showing that Leung teaches a method with a processor associated with a computer system performing the “associating” and “configuring” steps recited in claim 1.

On the record presented, we are not persuaded that Petitioner has shown a reasonable likelihood of prevailing on its assertion that claim 1, or claims 2–10 depending therefrom, are unpatentable over Leung.

3. Claims 11–15

Independent claim 11 is similar to claim 1, and recites a system comprising a “server computer,” “datastore,” “client computer,” plurality of “software application base objects,” plurality of “software application overlaid objects,” and “a runtime environment, stored in the datastore, comprising instructions that when executed cause the server to . . . receiv[e] a request from the client computer to execute a software application, the software application comprising a first application base object of the plurality of software application base objects.” Thus, similar to claim 1’s recitation of at least one base object “of” a software application that is configured by a processor to “execute,” claim 11 recites a software application “comprising” a base object and receiving a request for the software application to “execute.” Again, Petitioner cites portions of Leung referring to client application 300, master formset 18, and client formset 12, without clearly identifying what Petitioner is relying on as a “software

application” or explaining how any formset in the form of an XML file is a “software application” that “execute[s].” See Pet. 48–53 (discussing client application 300 and the “role defined Formset file” returned to client application 300); 37 C.F.R. §§ 42.104(b)(4)–(5). Thus, for the same reasons explained above regarding claim 1, we are not persuaded that Petitioner has shown a reasonable likelihood of prevailing on its assertion that claim 11, or claims 12–15 depending therefrom, are unpatentable over Leung.

4. Claims 16–20

Independent claim 16 is similar to claims 1 and 11, and recites a system comprising a “processor,” “datastore,” plurality of “software application base objects,” plurality of “software application overlaid objects,” and “a software application, stored in the datastore, configured for execution by the processor, comprising . . . a first software application base object.” Thus, similar to claim 11’s recitation of a software application “comprising” a base object and receiving a request for the software application to “execute,” claim 16 recites a software application “comprising” a base object and configured for “execution.” Petitioner includes little substantive argument for claim 16, primarily referring to its earlier discussion of claims 1 and 11. See Pet. 58–61. In its claim chart, Petitioner underlines portions of Leung describing server application 302 and the “role/category specific versions” of the definitions in master formset 18 after application of static overlays 400. *Id.* at 59–60. Petitioner does not clearly identify which of the two it is relying on as a “software application” or explain in any detail how either teaches the limitations of the claim. See *id.*; 37 C.F.R. §§ 42.104(b)(4)–(5). We also note that, like client application

300 on client 14, server application 302 on form server 16 does not “compris[e]” any XML definitions itself—it merely receives a formset request from client application 300, applies the overlays to master formset 18, and returns client formset 12. *See* Ex. 1003, col. 7, ll. 38–43. Thus, for the same reasons explained above regarding claims 1 and 11, we are not persuaded that Petitioner has shown a reasonable likelihood of prevailing on its assertion that claim 16, or claims 17–20 depending therefrom, are unpatentable over Leung.

IV. CONCLUSION

We conclude that Petitioner has not demonstrated a reasonable likelihood of prevailing with respect to at least one claim of the ’652 patent challenged in the Petition.

V. ORDER

In consideration of the foregoing, it is hereby:

ORDERED that the Petition is denied and no *inter partes* review is instituted.

IPR2018-00978
Patent 8,832,652 B2

PETITIONER:

Eugene Paige
KEKER, VAN NEST & PETERS LLP
epaige@kvn.com

PATENT OWNER:

Robert Auchter
Brett Cooper
Kevin Schubert
Christopher P. McNett
McKOOL SMITH, P.C.
rauchter@mckoolsmith.com
bcooper@mckoolsmith.com
kschubert@mckoolsmith.com
mcnett@gmail.com