UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
13/886,461	05/03/2013	Christian Bomhardt	000005-032500US	3369

58735 7590 02/04/2019

Fountainhead Law Group P.C.
Chad R. Walsh
900 LAFAYETTE STREET
SUITE 301
SANTA CLARA, CA 95050

EXAMINER	
MOBIN, HASANUL	

ART UNIT	PAPER NUMBER
2168	

NOTIFICATION DATE	DELIVERY MODE
02/04/2019	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

docketing@fountainheadlaw.com
rbaumann@fountainheadlaw.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte CHRISTIAN BOMHARDT

Appeal 2018-005714[1]
Application 13/886,461
Technology Center 2100

Before ST. JOHN COURTENAY III, LARRY HUME and JOYCE CRAIG, *Administrative Patent Judges*.

COURTENAY, *Administrative Patent Judge*.

DECISION ON APPEAL

Appellant appeals under 35 U.S.C. § 134(a) from the Examiner's Final rejection of claims 1, 3–9, 11–17, 19, and 20, which are all the claims pending in this application. Claims 2, 10, and 18 are cancelled. Final Act. 2. We have jurisdiction over the pending claims under 35 U.S.C. § 6(b).

We Reverse.

---

[1] Appellant identifies SAP SE as the real party in interest. App. Br. 2.

STATEMENT OF THE CASE

Introduction

Appellant's invention relates generally to "Performance And Quality Optimized Architecture For Cloud Applications." Spec. Title.

Exemplary Claim

1. A method for validating data comprising:

exchanging data between a server system and a client system;

providing a first data validation procedure expressed in a first specification language to the server system;

the server system causing the first data validation procedure to be executed on the server system to perform a validation of data exchanged between the server system and the client system in accordance with the first data validation procedure;

the server system providing the first data validation procedure that is expressed in the first specification language to the client system; and

the client system causing the first data validation procedure to be executed on the client system to perform a validation of data exchanged between the server system and the client system in accordance with the first data validation procedure,

wherein causing the first data validation procedure to be executed on a client system includes sending an interpreter engine and a representation of the first data validation procedure to the client system, wherein the interpreter engine executes on the client system to interpret the first data validation procedure.

(emphasis added regarding contested limitations).

Rejections

A. Claims 1, 3, 8, 9, 11, 16, 17, and 19 are rejected under 35 U.S.C. § 103 as being obvious over the combined teachings and suggestions of Kirkpatrick et al. (US 2005/0005163 A1; published Jan. 6, 2005) ("Kirkpatrick"), Colton et al. (US 8,756,579 B1; issued June 17, 2014) ("Colton"), and Moore et al. (US 6,915,454 B1; issued July 5, 2005) ("Moore"). Final Act. 4–9.

B. Claims 4–7, 12–15, and 20 are rejected under 35 U.S.C. § 103 as being obvious over the combined teachings and suggestions of Kirkpatrick, Colton, Moore, and Iborra et al. (US 2007/0089103 A1; published Apr. 19, 2007) ("Iborra"). Final Act. 9.

Issue on Appeal

Did the Examiner err in rejecting independent claims 1, 9, and 17 as being obvious over the cited combination of references under 35 U.S.C. § 103? In particular, we decide the question of whether the cited combination of Kirkpatrick, Colton, and Moore teaches or suggests the disputed "wherein" clause limitation,

> *wherein causing the first data validation procedure to be executed on a client system includes sending an interpreter engine and a representation of the first data validation procedure to the client system, wherein the interpreter engine executes on the client system to interpret the first data validation procedure*[,]

within the meaning of representative independent claim 1?[2] (emphasis added). We note the disputed "wherein" clause limitation is recited using

[2] We give the contested claim limitations the broadest reasonable interpretation (BRI) consistent with the Specification. *See In re Morris*, 127 F.3d 1048, 1054 (Fed. Cir. 1997).

similar language of commensurate scope in remaining independent claims 9 and 17.

Rejection of Independent Claim 1 under 35 U.S.C § 103

ANALYSIS

The Examiner finds "Kirkpatrick and Colton do not teach" the disputed "wherein clause" limitation. Final Act. 7 (emphasis added). The Examiner finds Moore provides the teachings missing from Kirkpatrick and Colton. In particular, the Examiner finds Moore teaches or suggests:

> wherein causing the *first data validation procedure* to be *executed on a client system* includes sending an *interpreter engine and a representation of the first data validation procedure* to the client system, wherein *the interpreter engine executes on the client system* to interpret the first data validation procedure (Moore, Fig. 16, Col 18, lines 37-61 illustrates operations relating to client-side web control validation. Moore discloses that saving operation saves input control object properties for transmission in an HTTP response. Rendering operation renders authoring language (e.g., HTML) data of the Web page for inclusion in the HTTP response. *Transmission operation sends the authoring language data to the client in an HTTP response which contains rendered validation script along with the control object (i.e., sending validation procedure with interpreter)*. Receipt operation receives input data through an input control element in the web page. Invocation operation invokes one or more validation scripts associated with the input control element to evaluate the input data against at least one validation criterion (i.e., the interpreter engine executes on the client system to interpret the first data validation procedure)).

Final Act. 7 (emphasis added).

Appellant contests the Examiner's findings regarding the "wherein" clause limitation recited in claim 1. Appellant explains the operation of Moore, in pertinent part:

> FIG. 16 (described beginning at column 18, line 37), for instance, illustrates operations relating to client-side web control validation. A close review of the process reveals, a rendering operation 1604 that renders authoring language (e.g., HTML) data for inclusion in an HTTP response, which includes one or more scripts for validating input data on the client. *Col. 18, lines 46 et seq.* A transmission operation 1606 sends the authoring language data to the client in the HTTP response. *Col. 18, lines 51 et seq.* The transmitted authoring language data, which contains rendered validation scripts, is received by the client. An invocation operation 1612 invokes one or more of the validation scripts to evaluate input data against at least one validation criterion. *Col. 18, lines 55 et seq.*

App. Br. 8–9.

Appellant specifically contends:

> *Moore does not describe "sending an interpreter engine and a representation of the first data validation procedure to the client system."* As explained above, Moore's server-side validation object can be used (i) to render authoring language data that is sent to the client in an HTTP response (in a server-side scenario), and (ii) to render appropriate client-side code to respond to a validation error (in a client-side scenario). *First, we note Moore does not disclose that the client-side code includes an interpreter engine.* Secondly, we further note Moore *does not disclose that the client-side code includes a representation of the first data validation procedure* (that is "executed on the server system to perform a validation of data").
>
> *Moore does not describe "wherein the interpreter engine executes on the client system to interpret the first data validation procedure."* The invocation operation described by Moore "invokes one or more validation scripts associated with the input control element to evaluate the input data against at least one validation criterion." Moore does not describe that invoking one or more validation scripts includes an interpreter engine that executes on the client system to interpret a first data validation procedure (that is "executed on the server system to perform a validation of data").

App. Br. 9 (emphasis added).

In response, the Examiner further explains the basis for the rejection.

> Moore, Fig. 16, Col 18, lines 37–61, illustrates operations relating to client-side web control validation. Moore discloses that saving operation saves input control object properties for transmission in an HTTP response. Rendering operation renders authoring language (e.g., HTML) data of the Web page for inclusion in the HTTP response. Transmission operation sends the authoring language data to the client in an HTTP response which contains rendered validation script along with *the control object (i.e., sending validation procedure with interpreter).*

Ans. 4 (emphasis added).

The Examiner further notes:

> Furthermore, Moore, Col 6, lines 51-67 discloses that each server-side control object in the control object hierarchy is called to generate (or render) data, such as HTML code, for display of client-side user interface elements in the web page. The term "render" (i.e., render () method) describe the operation of generating authoring language data that can be interpreted by [the] client-application, such as a browser, for display and client-side functionality.

Id.

Given the teachings in Moore (*id.*), the Examiner finds:

> Therefore, the transmitted authoring language data to the client from the server *contains rendered validation script* along with the control object (i.e., client received *authoring language data which is validation procedure with interpreter*). Similar to Validate() as described by the instant application specification paragraphs [0027-0028], *render() operation of Moore is considered as the interprete*r and the *data validation is performed when the client system invokes one or more validation scripts* associated with the input control element to *evaluate the input data against at least one validation criterion* (i.e., the interpreter engine executes on the client system to interpret the first data validation procedure).

Ans. 4 (emphasis added).

We particularly note the Examiner relies on Colton for teaching or suggesting the claimed "first data validation procedure, as contrasted with the claimed "a *representation* of the first data validation procedure" (claim 1), for which the Examiner relies on Moore, Fig. 16, col. 18, lines 37-61 — Final Act. 7. Specifically, the Examiner finds Colton teaches:

> providing *a first data validation procedure* expressed in a first specification language to the server system (A general method 100 of the present invention is shown in FIG. 8. At block 102, a single block of JavaScript code having a runat attribute of both is provided on a server-side (i.e., first data validation procedure expressed in a first specification language to the server system), Colton, Col 14, lines 54-56);

Final Act. 5.

Claim Construction

We begin our analysis with claim construction under the BRI standard, as applied to pending patent applications. *See supra*, *n.2*. Turning to the Specification for *context*, we observe that the literal claim term "interpreter engine" is not found in the main body of the Specification (not including the claims). However, literal support for the claim term "interpreter engine" is found in original dependent claims 2, 3, 10, 11, 18, and 19.[3] For example original claims 2 and 3 are reproduced below:

[3] "Although many original claims will satisfy the written description requirement" *Ariad Pharm., Inc. v. Eli Lilly & Co.* (598 F.3d 1336, 1349 (Fed. Cir. 2010) (*en banc*)), the Patent Rules require:

> [t]he claim or claims must conform to the invention as set forth in the *remainder of the specification* and the terms and phrases used in the claims *must find clear support or antecedent basis in*

Original claim 2: "The method of claim 1 wherein causing the first data validation procedure to be executed on a client system includes sending an *interpreter engine* and a representation of the first data validation procedure to the client system, wherein the *interpreter engine* executes on the client system to interpret the first data validation procedure." (emphasis added).

Original claim 3: "The method of claim 2 wherein the *interpreter engine* and the *first data validation procedure* are sent to the client system in *a web page*." (emphasis added). Paired original dependent claims (10, 11) and (18, 19) recite similar language having commensurate scope.

In similar form, dependent claim 3 (before us on appeal) recites: "The method of claim 1 wherein the *interpreter engine* and the *first data validation procedure* are sent to the client system in a *web page*." (emphasis added). (Note: original claim 2 was cancelled during prosecution).

Thus, as an initial matter of claim construction, the doctrine of claim differentiation informs us of the presumption that the broader independent claims 1, 9, and 17 before us on appeal do not require that (one or both of) the *interpreter engine* and the *first data validation procedure* must be sent from the server to the client system via a *web page*.[4]

> *the description* so that the meaning of the terms in the claims may be ascertainable *by reference to the description*.

37 C.F.R. § 1.75(d)(1) (emphasis added).

[4] *See Phillips v. AWH Corp.*, 415 F.3d 1303, 1314–15 (Fed. Cir. 2005) (en banc) ("Differences among claims can also be a useful guide in understanding the meaning of particular claim terms. For example, the presence of a dependent claim that adds a particular limitation gives rise to a presumption that the limitation in question is not present in the independent

Turning to Appellant's Specification (including the drawings), we find descriptions (¶¶ 27–28) and drawings (e.g., Fig. 4) which indicate the claimed "*representation* of the first data validation procedure" corresponds to Appellant's Figure 4, element 402: i.e., the declaration of the array: "var valArray =['ZIP' ', 'NUM', '5']; //dynamically inserted by web server."

Similarly, the claimed "interpreter engine" corresponds to Appellant's Figure 4, element 404, i.e., "function Validate () // Generic validate function." [5]

In particular, see paragraphs 27 and 28 of Appellant's Specification, describing Appellant's Figure 4, as reproduced below in pertinent part:

> [0027] . . . The HTML code 400 includes a section 402 that *expresses the data validation procedure* 104 shown in Table 3 *as data in an array* called valArray[]. A section 404 defines an *interpreter function* called Validate(), which comprises code (not shown) that can interpret or otherwise execute the rules specified in valArray[]. A section 406 represents the body of the HTML code. The body invokes the Validate() function when a mouse click event is detected in order to validate the input data.
>
> [0028] The approach illustrated in Fig. 4 uses an interpreter, namely Validate(), to process an array (e.g., valArray[]) that contains the rules set forth in the first data validation procedure 104. The array valArray[] may be initialized by the server

---

claim.") (citations omitted); *see also Uniroyal, Inc.* v. *Rudkin-Wiley Corp.*, 837 F.2d 1044, 1054-55 (Fed. Cir. 1988); *Liebel-Flarsheim Co. v. Medrad, Inc.*, 358 F.3d 898, 910 (Fed. Cir. 2004).

[5] *See* "Summary of Claimed Subject Matter" section of the Appeal Brief: "sending an *interpreter engine* and a *representation of the first data validation procedure* (404, 402, FIG. 4, ¶ [0027]) to the client system, wherein the interpreter engine executes on the client system to interpret the first data validation procedure (¶ [0028])." App. Br. 5.

> system 112 with coding from the first data validation procedure 104. Data validation is performed when the client system 114 invokes the interpreter function Validate()to run through the contents of the array valArray[].

(emphasis added).

Figure 4 of Appellant's drawings is reproduced below:

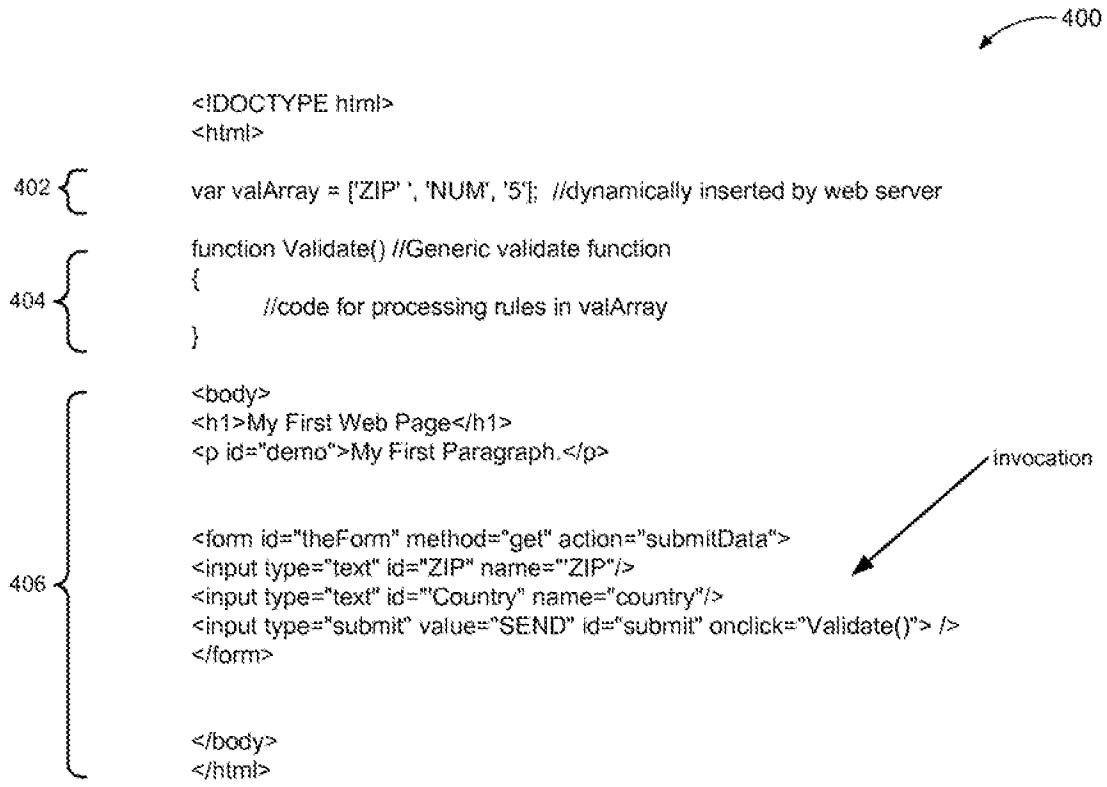

Fig. 4

Figure 4 of Appellant's drawings is reproduced above, depicting: (1) the claimed "*representation of the first data validation procedure*" as element 402, i.e., corresponding to the *declaration* of an array "valArray" and, (2) the claimed "*interpreter engine*" as element 404, i.e., a generic function Validate (), as recited in independent claims 1, 9 and 17 before us on appeal. (emphasis added).

We particularly observe that claim construction has not been developed in the record, either by the Examiner or by the Appellant. We emphasize that claim construction is an important step in a patentability determination. A legal conclusion that a claim is obvious involves two analytical steps, assuming the references have been properly combined under § 103. *See Medichem, S.A. v. Rolabo, S.L.*, 353 F.3d 928, 933 (Fed. Cir. 2003) ("Both anticipation under § 102 and obviousness under § 103 are two-step inquiries. The first step in both analyses is a proper construction of the claims. . . . The second step in the analyses requires a comparison of the properly construed claim to the prior art." (internal citations omitted)). Under the second step, the Board must compare the construed claim to one or more prior art references and make factual findings regarding the limitations contested by Appellant. *See In re Crish*, 393 F.3d 1253, 1256 (Fed. Cir. 2004).

Claim Construction for "interpreter engine" (Claim 1)

Under a broad but reasonable interpretation of Appellant's claim 1, (including Figure 4 and the supporting description found in paragraphs 27 and 28 of Appellant's Specification), we conclude the recited "interpreter engine" encompasses *any* data validation method, function, or procedure, as implemented in *any* programming language or script that is an interpreted (as opposed to compiled) language or script. Moreover, we note that the "wherein" clause of claim 1 expressly requires the "interpreter engine" to execute *on the client system* to interpret the "first data validation procedure" that is also intended "to be executed on a client system." Claim 1. [6]

[6] Our reviewing court guides that "[i]nterpretation of descriptive statements in a patent's written description is a difficult task, as an inherent tension

Claim Construction for "representation of the first data validation procedure" (Claim 1)

Under a broad but reasonable interpretation of Appellant's claim 1, (including Figure 4 and the supporting description found in paragraphs 27 and 28 of the Specification), we conclude the recited "*representation* of the first data validation procedure" is not the same as the *actual* "first data validation procedure" interpreted code or script, but instead merely "represents" it in *any* manner consistent with the Appellant's Specification. (emphasis added).

For example, the recited "*representation* of the first data validation procedure" could correspond, *inter alia*, to an associated declaration (as depicted in element 402 of Appellant's Figure 4 – "var valArray = ['ZIP' ', 'NUM', '5'];"), or more broadly, as another example, as a function pointer (i.e., pointing to the address of the "first data validation procedure").

---

exists as to whether a statement is a clear lexicographic definition or a description of a preferred embodiment. The problem is to interpret claims 'in view of the specification' without unnecessarily importing limitations from the specification into the claims." *E-Pass Techs., Inc. v. 3Com Corp.*, 343 F.3d 1364, 1369 (Fed. Cir. 2003) (citation omitted).

Mapping of the properly construed claim terms to the corresponding features found in Moore under 37 C.F.R. § 1.104(c)(2)[7]

> Given the teachings in Moore (*id.*), the Examiner finds:
>
> Therefore, the transmitted authoring language data to the client from the server contains rendered validation script along with the control object (i.e., client received *authoring language data which is validation procedure with interpreter*). Similar to Validate() as described by the instant application specification paragraphs [0027-0028] [of Appellant's Specification], *render() operation of Moore is considered as the interpreter* and the data validation is performed when the client system invokes *one or more validation scripts* associated with the input control element to evaluate the input data against at least one validation criterion (i.e., the interpreter engine executes on the client system to interpret the first data validation procedure).

Ans. 4 (emphasis added).

From the above explanation in the Answer (4), the Examiner is reading the *"interpreter engine"* recited in claim 1 on Moore's render () operation (col. 18, ll. 46–51).

From the above explanation in the Answer (4), it appears that the Examiner may be reading the *"***representation** *of the first data validation procedure"* recited in claim 1, on Moore's transmitted authoring language (i.e., HTML web page) that *contains* a rendered validation script, wherein the Examiner appears to be cumulatively reading the "first data validation procedure" recited in claim 1 on Moore's validation script. *See* Moore, (col. 18, ll. 55–61). As discussed above, the Examiner principally relies on

---

[7] *See* 37 C.F.R. § 1.104(c)(2) ("When a reference is complex or shows or describes inventions other than that claimed by the applicant, the particular part relied on must be designated as nearly as practicable. The pertinence of each reference, if not apparent, must be *clearly explained* and each rejected claim specified.") (emphasis added).

Colton for teaching or suggesting the claimed "first data validation procedure." Claim 1, Final Act. 5.

We now turn to Moore's Figure 16, in which the rendering operation 1604 is described as:

> render[ing the] authoring language (e.g., HTML) data of the Web page for inclusion in the HTTP response. In contrast to the serverside validation description, however, the rendered HTML *includes one or more scripts for validating input data* on the client. Transmission operation 1606 sends the *authoring language data* to the client in an HTTP response. Termination operation 1608 terminates the page and control objects on the server.

Moore col. 18, ll. 46–54; *see also* Fig. 16.

We note that if the rendered HTML web page (i.e., the authoring language data sent from the server to the client) includes a validating script (i.e., an "interpreter engine" validation script), then the authoring language data itself (i.e., the rendered HTML web page sent from the server to the client) might be considered "representative" of the claimed "first data validation procedure." Claim 1.

However, we find the Examiner has not fully developed the record to show this, if that was the Examiner's intended mapping of the disputed claim terms to the corresponding features found in Moore. Moreover, such a reading might be considered by our reviewing court to be unreasonable, as being overly broad and inconsistent with the description of array declaration "valArray" element 402 (Fig. 4). *See* Spec. ¶ 27.

Based upon our review of the record, we find the Examiner does not explain in sufficient detail the aforementioned mapping of the disputed claim terms to the corresponding specific features found in Moore. Moreover, we find the record created by the Examiner could be subject to

alternative mappings that depart from our understanding of the record, as discussed above. *Cf.* "The Examiner's discussion on page 4 of the Answer equates Moore's 'control object' with the claimed "interpreter engine." Reply Br. 2. It is apparent to us that Appellant is also unsure as to the Examiner's intended mapping.

In setting forth the rejection of claim 1, we find the Examiner paints with a broad brush. Final Act. 7. The Examiner's proffered mapping of the disputed claim limitations to the corresponding features found in Moore (*id.*) is imprecise. Therefore, we find that to affirm the Examiner on this record would require us to engage in some degree of speculation. We decline to engage in speculation.[8]

Moreover, we note that Moore describes the operation of the rendering function in detail:

> In operation 208, each server-side control object in the control object hierarchy is called to generate (or render) data, such as HTML code, for display of client-side user interface elements in the web page. Note that, although the term "render" may be used to describe the operation of displaying graphics on a user interface, the term "render" is also used herein to describe the operation of generating *authoring language data* that can be *interpreted by client application, such as a browser*, for display and client-side functionality. In one embodiment, calls to render( ) methods in individual control objects are performed using a tree traversal sequence. That is, *a call to the renders method of a page object results in recursive traversal throughout appropriate server-side control objects* in the

[8] "A rejection . . . must rest on a factual basis." *In re Warner*, 379 F.2d 1011, 1017 (CCPA 1967). "The Patent Office has the initial duty of supplying the factual basis for its rejection. It may not . . . resort to speculation, unfounded assumptions or hindsight reconstruction to supply deficiencies in its factual basis." *Id.*

> control object hierarchy. Alternative methods for calling the render() methods for appropriate control objects may also be employed, including an event signaling or object registration approach. The parentheses designate the "render( )" label as indicating a method, as compared to a data property.

Moore Col. 6, l.51—col. 7, l.2 (emphasis added).

From the above description (*id.*), the control objects are described as being located on the server-side, and (according to one embodiment) it is the client-side browser (or other client side application) that does the interpreting of the "authoring language data." *Id.*

Based upon our review of the record, we find *a preponderance of the evidence* supports Appellant's contentions for essentially the same reasons argued in the Briefs, as discussed above. Therefore, we are constrained by the record before us to find the Examiner erred in concluding that the cited combination of Kirkpatrick, Colton, and Moore renders obvious Appellant's independent claim 1.

Accordingly, we reverse the Examiner's § 103 Rejection A of independent claim 1, and for the same reasons we also reverse Rejection A of remaining independent claims 9 and 17, which recite the contested limitations using similar language of commensurate scope. Because we have reversed the rejection of all independent claims on appeal, for the same reasons, we reverse the Examiner's Rejection A of the associated dependent claims also rejected under Rejection A.

Rejection B

Regarding the remaining dependent claims 4–7, 12–15, and 20, rejected under § 103 Rejection B, the Examiner has cited an additional secondary reference, Iborra, as evidence in support of the rejection. On this record, we find the Examiner has not shown how Iborra overcomes the

deficiencies of the base combination of Kirkpatrick, Colton, and Moore, as discussed above regarding Rejection A of independent claims 1, 9, and 17. Therefore, we are constrained on this record to also reverse Rejection B of the remaining dependent claims 4–7, 12–15, and 20.

Conclusion

The Examiner erred in rejecting claims 1, 3–9, 11–17, 19, and 20 as being obvious over the cited combination of references under 35 U.S.C. § 103.

DECISION

We reverse the Examiner's decision rejecting claims 1, 3–9, 11–17, 19, and 20 under 35 U.S.C. § 103(a).

REVERSED