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.
12/948,113	11/17/2010	Shreyank Gupta	05220.871 (P0824)	3217

14400 7590 08/31/2020
LOWENSTEIN SANDLER LLP / Red Hat
Patent Docket Administrator
One Lowenstein Drive
Roseland, NJ 07068

EXAMINER	
FISHER, PAUL R	

ART UNIT	PAPER NUMBER
3689	

NOTIFICATION DATE	DELIVERY MODE
08/31/2020	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):

patents@lowenstein.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte SHREYANK GUPTA

Appeal 2019-006439
Application 12/948,113
Technology Center 3600

Before JEFFREY S. SMITH, MICHAEL J. STRAUSS, and MICHAEL M. BARRY, *Administrative Patent Judges*.

BARRY, *Administrative Patent Judge*.

DECISION ON APPEAL[1]

Pursuant to 35 U.S.C. § 134(a), Appellant[2] appeals from the Examiner's decision to reject claims 1–6, 8–13, and 15–19, which are all of the pending claims. We have jurisdiction under 35 U.S.C. § 6(b).

We affirm.

[1] We refer herein to the Appeal Brief ("Appeal Br.") filed May 15, 2018, the Answer ("Ans.") mailed Sept. 7, 2018, the Reply Brief filed Dec. 20, 2018 ("Reply Br."), the Final Office Action ("Final Act.") mailed Sept. 26, 2017, and the original Specification ("Spec.") and Figures ("Figs.") filed Nov. 17, 2010.

[2] We use "Appellant" to refer to "applicant" as defined in 37 C.F.R. § 1.42. Appellant identifies the real party in interest as Red Hat, Inc. Appeal Br. 3.

Introduction

Appellant describes that "[e]mbodiments of the present invention relate to project management [and s]pecifically . . . to automatically modifying task status in a project management system via keywords in source code version control commit logs." Spec. ¶ 1. In discussing the background to the invention, Appellant explains that, in the situation of a task status being tracked both by a project management tool and a source code revision control tool, when the project management tool requires manual updating of task status data whereas the source code revision control tool automatically updates its corresponding task status data, "the task status data in the project management tool is not always up to date and reliable." Spec. ¶ 5.

Claims 1, 8, and 15 are independent. Claim 1 is illustrative:

> 1. A method comprising:
>
> identifying a plurality of tasks of modifying source code for a project and corresponding current task status data for the plurality of tasks in a project management data store coupled to a project management system, wherein the current task status data indicates a current state of modifying the source code for a corresponding task;
>
> extracting, from a directory location of a source code revision control system separate from the project management system, a commit log indicating changes made to the source code, the commit log stored by the source code revision control system in a source code revision data store separate from the project management data store, and the commit log comprising new task status data for one of the plurality of tasks;
>
> determining whether the commit log has been previously parsed;

> in response to a determination that the commit log has not been previously parsed, identifying a keyword within the commit log;
>
> parsing, by a processing device, data in the commit log that is located within a defined range of the keyword to identify a task identifier in the commit log that corresponds to the keyword, wherein the keyword comprises the new task status data that indicates a new state of modifying the source code for the one of the plurality of tasks;
>
> changing the current task status data in the project management data store for the one of the plurality of tasks to reflect the new state of modifying the source code for the one of the plurality of tasks to synchronize the current task status data in the project management data store with the new state of modifying the source code as indicated in the commit log extracted from the source code revision control system;
>
> continuously updating the current task status data in the project management data store in view of a subsequent parsing of the commit log;
>
> transmitting a notification via a network to a client device, the notification reflecting the current task status data as changed in the project management data store; and
>
> in response to a determination that the keyword is not located in the commit log, deleting the commit log.

Appeal Br. 22–23 (Claims App'x).

References and Rejections

The rejections rely on the following references:

Name	Reference	Date
Oddo	US 2003/0105681 A1	June 5, 2003
Guturu	US 6,581,075 B1	June 17, 2003
Wakui	US 2005/0096934 A1	May 5, 2005
McKnight	US 7,143,345 B2	Nov. 28, 2006
Chigusa	US 2008/0154882 A1	June 26, 2008
Motoyama	US 7,406,432 B1	July 29, 2008
Best	US 8,312,430 B2	Nov.13, 2012

The Examiner rejected claims 1–6, 8–13, and 15–19 (i.e., all pending claims) under 35 U.S.C. § 101 as directed to a judicial exception (i.e., an abstract idea), without reciting significantly more. Final Act. 2–4.

The Examiner rejected claims 1, 2, 4–6, 8, 9, 11–13, 15, 16, 18, and 19 under 35 U.S.C. § 103(a) as unpatentable over Motoyama, Best, Oddo, McKnight, Chigusa, and Wakui. Final Act. 4–56.

The Examiner rejected claims 3, 10, and 17 under 35 U.S.C. § 103(a) as unpatentable over Motoyama, Best, Oddo, McKnight, Chigusa, Wakui, and Guturu. Final Act. 56–64.

ANALYSIS

The § 101 Rejection

For the § 101 rejection, Appellant argues all pending claims together as a group. Appeal Br. 7–11. We select claim 1 as representative. *See* 37 C.F.R. § 41.37(c)(1)(iv).

In issues involving subject matter eligibility, our inquiry focuses on whether the claims satisfy the two-step test set forth by the Supreme Court in *Alice Corp. v. CLS Bank Int'l*, 573 U.S. 208 (2014). The Court instructs us to "first determine whether the claims at issue are directed to a patent-ineligible concept." *Id.* at 218. In this case, the inquiry centers on whether the claims are directed to an abstract idea. If the initial threshold is met, we then move to the second step, in which we "consider the elements of each claim both individually and 'as an ordered combination' to determine whether the additional elements 'transform the nature of the claim' into a patent-eligible application." *Id.* at 217–18 (quoting *Mayo Collaborative Servs. v. Prometheus Labs., Inc.*, 566 U.S. 66, 79, 78 (2012)). The Court describes the second step as a search for "an '"inventive concept"'—*i.e.*, an element or combination of elements that is 'sufficient to ensure that the patent in practice amounts to significantly more than a patent upon the [ineligible concept] itself.'" *Id.* (quoting *Mayo*, 566 U.S. at 72–73).

In 2019, the USPTO published revised guidance on the application of § 101 consistent with *Alice* and subsequent Federal Circuit decisions. *See* 2019 Revised Patent Subject Matter Eligibility Guidance, 84 Fed. Reg. 50–57 (Jan. 7, 2019) ("Guidance") *see also* October 2019 Patent Eligibility Guidance Update, 84 Fed. Reg. 55942–53 (Oct. 17, 2019) (providing "examples as well as a discussion of various issues raised by the public comments" to the Guidance). Under the Guidance, we first look to whether the claim recites:

> (1) any judicial exceptions, including certain groupings of abstract ideas (i.e., mathematical concepts, certain methods of organizing human activity such as a fundamental economic practice, or mental processes) (referred to as Step 2A, prong 1 in the Guidance); and
>
> (2) additional elements that integrate the judicial exception into a practical application (*see* MPEP § 2106.05(a)–(c), (e)–(h)) (referred to as Step 2A, prong 2 in the Guidance).

See Guidance, 84 Fed. Reg. at 52–55.

Only if a claim (1) recites a judicial exception and (2) does not integrate that exception into a practical application, do we then move to Step 2B of the Guidance, in which we look to whether the claim:

> (3) adds a specific limitation beyond the judicial exception that is not "well-understood, routine, conventional" in the field (*see* MPEP § 2106.05(d)); or
>
> (4) simply appends well-understood, routine, conventional activities previously known to the industry, specified at a high level of generality, to the judicial exception.

See Guidance, 84 Fed. Reg. at 56.

Alice/Mayo Step One, 2019 Guidance Step 2A, Prong One
(Does Claim 1 Recite a Patent-Ineligible Concept?)

For our prong one analysis, we consider the following limitations shown here *in italics* to be the only limitations in claim 1 that do not describe (recite) a part of an abstract idea:

> [1] identifying a plurality of tasks of modifying source code for a project and corresponding current task status data for the plurality of tasks in a project management *data store coupled to a* project management *system*, wherein the current task status data indicates a current state of modifying the source code for a corresponding task;
>
> [2] extracting, from *a directory* location of a source code revision control system separate from the project management system, a commit log indicating changes made to the source code, the commit log stored by the source code revision control system in a source code revision *data store separate from the project management data store*, and the commit log comprising new task status data for one of the plurality of tasks;
>
> [3] determining whether the commit log has been previously parsed;
>
> [4] in response to a determination that the commit log has not been previously parsed, identifying a keyword within the commit log;
>
> [5] parsing*, by a processing device,* data in the commit log that is located within a defined range of the keyword to identify a task identifier in the commit log that corresponds to the keyword, wherein the keyword comprises the new task status data that indicates a new state of modifying the source code for the one of the plurality of tasks;
>
> [6] changing the current task status data in the project management *data store* for the one of the plurality of tasks to reflect the new state of modifying the source code for the one of the plurality of tasks to synchronize the current task status data

> in the project management data store with the new state of modifying the source code as indicated in the commit log extracted from the source code revision control system;
>
> [7] continuously updating the current task status data in the project management data store in view of a subsequent parsing of the commit log;
>
> [8] transmitting a notification *via a network* to a client *device*, the notification reflecting the current task status data as changed in the project management data store; and
>
> [9] in response to a determination that the keyword is not located in the commit log, *deleting* the commit log.

The first step recites "identifying a plurality of tasks of modifying source code for a project and corresponding current task status data for the plurality of tasks in [] project management data . . . [for] a project management system,[3] wherein the current task status data indicates a current state of modifying the source code for a corresponding task." This describes an activity that human source code developers can perform in their minds, e.g., an observation or evaluation. Thus, the first step recites an abstract idea in the category of mental processes. *See* 2019 Guidance, 84 Fed. Reg. at 52. The first step also recites an abstract concept because it describes a way to manage human interactions (e.g., among source code developers) following rules or instructions. The 2019 Guidance explains this is of one of certain

[3] Limitations describing the relatedness of "project management data" to "a project management system" are abstract because, as discussed *infra*, people can establish these data relationships (1) using mental processes or (2), in the course of following rules or instructions during managed interactions between people, i.e., as part of one of certain methods of organizing human activity; in either case, as the 2019 Guidance explains, such limitations describe (recite) an abstract idea. 84 Fed. Reg. at 52.

methods of organizing human activity that, in accord with judicial precedent, are deemed abstract. *Id.*

The second step recites "extracting, from a [] location of a source code revision control [SCRC] system [SCRCS] separate from the project management [PM] system [PMS], a commit log indicating changes made to the source code, the commit log stored by the source code revision control system . . ., and the commit log comprising new task status data for one of the plurality of tasks." This describes obtaining the "commit log" with "new task status data" from a "separate" SCRCS location. Subsequent steps parse the commit log to obtain data and use it to synchronize PMS task status data with the SCRCS task status data. Except for generic computer requirements such as a "processing device" or "data store," human source code developers can perform all of the functionality of these steps in their minds and/or using pen and paper (i.e., mental steps) or by interacting as instructed (i.e., managing personal behavior or interactions). For example, developers can be instructed to observe whether a commit log file has a modification date this is later than the date of the last time the file was parsed, thus "determining whether the commit log has been previously parsed," as recited. Thus, the foregoing limitations of claim 1 recite abstract concepts because they amount to mental processes or rules or instructions for managing human interactions, i.e., one of certain methods of organizing human activity that have been deemed to be abstract as described in the 2019 Guidance. *Id.*

Accordingly, claim 1 recites a judicial exception in the form of an abstract idea for, in the words of Appellant's Specification, "automatically modifying task status in a project management system via keywords in source code version control commit logs." Spec. ¶ 1. Stated differently, claim 1 recites an abstract idea for using SCRC commit logs to synchronize task status data between PM and SCRC systems. *See Apple, Inc. v. Ameranth Inc.*, 842 F.3d 1229, 1240 (Fed. Cir. 2016) ("An abstract idea can generally be described at different levels of abstraction."). Thus, our analysis proceeds to prong two.

Alice/Mayo Step One, 2019 Guidance Step 2A, Prong Two
(Does Claim 1 Integrate the Abstract Idea into a Practical Application?)

We next consider whether the claim integrates the abstract idea into a practical application. 2019 Guidance, 84 Fed. Reg. at 54. To determine this, we identify whether there are "*any additional elements recited in the claim beyond the judicial exception(s)*" and evaluate those elements to determine whether they integrate the judicial exception into a practical application. 84 Fed. Reg. at 54–55 (emphasis added); *see also* MPEP § 2106.05(a)–(c), (e)–(h).

Here, as identified above, beyond the limitations describing the abstract idea, claim 1 recites the following technological limitations, shown here *in italics*: "a project management *data store coupled to a* project management *system*," "a source code revision *data store separate from the project management data store*," extracting the commit log from "*a directory* location," "parsing, *by a processing device*," "*continuously* updating the current task status data," and "*deleting* the commit log." "Data stores," "processing devices," "continuously" performing tasks, "directory" locations, and "deleting" are basic features of fundamental computer system technology. The Specification confirms this by describing such technology features at a high, generic level. *See, e.g.*, Spec. ¶¶ 19–23, 29, 37, 52. These additional elements are generic and do not result in an improvement to a technology or technical field—instead, the claim recites basic technology to automate performance of an abstract idea. There is no improvement to "the functioning of the computer itself" or "any other technology or technical field." *See* MPEP § 2106.05(a) (quoting *Alice*, 573 U.S. at 225). Neither do these computer limitations qualify as applying the judicial exception with "a particular machine," because the "computer system" provides its conventional functions and requires no more than general purpose computer equipment. *See* MPEP § 2106.05(b); *see also Ultramercial, Inc. v. Hulu, LLC*, 772 F.3d 709, 716–17 (Fed. Cir. 2014); *In re TLI Commc'ns LLC Patent Litigation*, 823 F.3d 607, 613 (Fed. Cir. 2016) (explaining that mere recitation of concrete or tangible components is not an inventive concept).

Appellant's arguments similarly do not persuade us that claim 1 effects a transformation of any recited articles, which are simply used for their ordinary purposes, or that claim 1 includes any other meaningful (technological) limitations, i.e., limitations beyond simply "linking the use" of the abstract idea to generic technology. *See* MPEP § 2106.05 (c), (e)–(f); *see also id.* at (g)–(h) (use of well-known limitations beyond the judicially excepted matter constitutes "insignificant extra-solution activity" (g) and claim limitations "merely indicating a field of use or technological environment in which to apply a judicial exception do not amount to significantly more" (h)).

Accordingly, we determine the recited judicial exception is not integrated into a practical application, and that the Examiner did not err in determining claim 1 is directed to an abstract idea. Accordingly, we proceed to step two of the *Alice/Mayo* analysis (2019 Guidance Step 2B).

Alice/Mayo Step Two, 2019 Guidance Step 2B
(Does Claim 1 Recite Significantly More than the Abstract Idea?)

In step two of the *Alice/Mayo* analysis, we consider whether there are additional limitations that individually, or as an ordered combination, ensure the claims amount to "significantly more" than the abstract idea. *Alice*, 573 U.S. at 217–18 (citing *Mayo*, 566 U.S. at 72–73, 77–79). As the 2019 Guidance explains, many of the considerations to determine whether a claim amounts to "significantly more" under step two of the *Alice* framework are already considered as part of determining whether the judicial exception has been integrated into a practical application. 84 Fed. Reg. at 56. Thus, at this point of our analysis, we determine if claim 1 adds a specific limitation, or combination of limitations, that is not a well-understood, routine, conventional activity in the field; or whether it simply recites well-understood, routine, conventional activities at a high level of generality. *Id.*

Here, as the Examiner finds, and we agree, claim 1 does not recite limitations (or a combination of limitations) additional to those for the abstract idea that are beyond what were known to those of ordinary skill in the art to be well-understood, routine, and conventional prior to the invention. Final Act. 3–4. The high-level, generic disclosure of computer technology in Appellant's Specification related to claim 1 confirms this finding. *See, e.g.*, Spec. ¶¶ 13–23, 29, 37.

Section 101 Conclusion

Appellant's arguments of error in the § 101 rejection are unpersuasive in view the foregoing. For clarity, we discuss selected arguments. For example, in support of the argument "the claims include specific recitations that may improve interoperability with a third-party system and data synchronization that are not abstract and are patentable," Appellant compares claim 1 to the claim at issue in in *Aatrix Software, Inc. v. Green Shades Software, Inc.*, 882 F.3d 1121 (Fed. Cir. 2018); *see also* Order on Petition for Rehearing En Banc, 890 F.3d 1354 (Fed. Cir. 2018). Appeal Br. 15; Reply Br. 21. Appellant points to paragraph 12 of the Specification as "indicat[ing] that the claims improve the reliability of the task status data in the project management system using the disclosed techniques." Reply Br. 20. This argument is unpersuasive. Paragraph 12 of the Specification explains:

> Typically, whenever a source code revision control system performs a commit, a user has to manually change the task statuses in the project management data for the project. Until this manual process is performed, the task status data in the source code revision control system and the task status data in the project management system may not be in sync. Embodiments of the present invention provide a system to automate this manual process. Users can use a source code revision control system to commit source code changes and embodiments of the invention automatically can update the task status in a project management system via keywords in the source code revision control commit logs to provide more reliable data in the project management tool.

Spec. ¶ 12. This describes improvements as arising from automating a previously manual process, not from a technological improvement to technology for data synchronization and interoperability. Specification paragraph 12 supports the determination that the improvement arises from the recited abstract idea(s). *See, e.g.*, *Bancorp Servs., L.L.C. v. Sun Life Assur. Co. of Canada (U.S.)*, 687 F.3d 1266, 1277–79 (Fed. Cir. 2012) ("Using a computer to accelerate an ineligible mental process does not make that process patent-eligible.”); *SiRF Tech., Inc. v. Int’l Trade Comm’n*, 601 F.3d 1319, 1333 (Fed. Cir. 2010) (“In order for the addition of a machine to impose a meaningful limit on the scope of a claim, it must play a significant part in permitting the claimed method to be performed, rather than function solely as an obvious mechanism for permitting a solution to be achieved more quickly.”). Moreover, Appellant’s reliance on *Aatrix* is ill-founded as the court cautions that its holding and dicta are specifically limited to the procedural context of Rule 12(b)(6) of the Federal Rules of Civil Procedure. *Aatrix*, 882 F.3d at 1129–30 (“Nothing in this opinion should be viewed as going beyond the Rule 12(b)(6) stage.”). Appellant proffers no explanation for how the situation at hand is akin to *Aatrix*’s context of a Rule 12(b)(6) motion.

Appellant also contends *Enfish, LLC v. Microsoft Corp.*, 822 F.3d 1327 (Fed. Cir. 2016), compels a determination that claim 1 is patent eligible. Appeal Br. 15–17; *see also* Reply Br. 22, 31–32. This argument is unpersuasive because, unlike the claim at issue in *Enfish*, which recited a specific database technology improvement, *see* 822 F.3d at 1335–37, Appellant’s claim 1 recites only routine technology to automate an otherwise abstract process.

Appellant’s reliance on *Amdocs (Israel) Limited v. Openet Telecom, Inc.*, 841 F.3d 1288 (Fed. Cir. 2016), is similarly unavailing. *See* Appeal Br. 19–20; Reply Br. 24. The claim at issue in *Amdocs* recited a combination of “arguably generic components” but also recited a specific “enhancing limitation [that] necessarily requires that these generic components operate in an unconventional manner to achieve an improvement in computer functionality.” 841 F.3d at 1300–01. Unlike *Amdocs’* specific recitation of a limitation necessarily requiring unconventional operation of generic technology components, claim 1, as discussed above, recites only conventional operation of technology components, in order to automate an otherwise manual process.

Appellant's arguments based on *BASCOM Global Internet Services, Inc. v. AT&T Mobility LLC*, 827 F.3d 1341 (Fed. Cir. 2016), are also not persuasive. *See* Appeal Br. 20–21; Reply Br. 24–25. The automation of a previously manual task to improve synchronizing task status data across two software packages is not analogous to the claims at issue in *BASCOM*, which recited a combination including an ISP server, remote client devices, and particular Internet filtering elements or schemes placed within that network environment to provide a specific technological improvement to the state of the art for Internet filtering. *See BASCOM*, 827 F.3d at 1345–46 (setting forth claims), 1349–50 (finding the patent eligible claims "recite a specific, discrete implementation of the abstract idea of filtering content" and do not "preempt all ways of filtering content on the Internet").

Accordingly, we sustain the § 101 rejection of claim 1. We also, therefore, sustain the § 101 rejection of claims 2–6, 8–13, and 15–19, which Appellant argues are patent eligible for the same reasons as claim 1. *See* Appeal Br. 14–21. In doing so, as consistent with the above, we adopt as our own the Examiner's findings and reasoning as set forth in the Final Rejection and Answer.

The § 103 Rejections

Appellant argues Examiner error in the rejection of all pending claims based on independent claim 1 and separately argues error in the rejection of dependent claims 3, 10, and 17 based on claim 3. *See* Appeal Br. 7–14. We have reviewed the Examiner's rejections of claims 1 and 3 and disagree with Appellant's conclusions. Instead, as consistent with our discussion below, we adopt the Examiner's findings and reasons as set forth in the Final Office Action from which this appeal is taken and as set forth in the Answer. We highlight the following for emphasis.

Claim 1

Claim 1 recites

> extracting, from a directory location of a [SCRCS] separate from the [PMS], a commit log indicating changes made to the source code, the commit log stored by the [SCRCS] in a source code revision data store separate from the project management data store, and the commit log comprising new task status data for one of the plurality of tasks.

The Examiner finds Motoyama teaches or suggests this limitation except for the recited aspect of the SCRCS being separate from the PMS, for which the Examiner relies on the teachings of Wakui in combination with Motoyama (and the remaining references). Final Act. 5–6, 16 (citing Motoyama 7:15–51, 10:2–32, 12:11–45, Figs. 5B, 12–13; Wakui ¶¶ 23, 31, 47, 52, 62). Appellant contends the Examiner errs in this finding because Motoyama does not teach or suggest the recited requirement "that the log is ***stored by a [SCRCS] in a source code revision data store separate from the project management data store***." Appeal Br. 9 ("There does not appear to be a teaching or suggestion in Motoyama of extracting a commit log from a [SCRCS] that is separate from a [PMS].").

This argument, which focuses on the disclosure of Motoyama while ignoring the disclosure of Wakui, is unpersuasive. The Examiner explained that the combination of Motoyama and Wakui, not Motoyama alone, teaches this disputed limitation. Final Act. 16. The relevant inquiry is whether the claimed subject matter would have been obvious to those of ordinary skill in the art in light of the *combined teachings* of those references. *In re Keller*, 642 F.2d 413, 426 (CCPA 1981).

Appellant further argues "Motoyama does not teach or suggest a source code revision control system at all." Reply Br. 6. Appellant construes Motoyama too narrowly. Motoyama discloses that objects in its project database can include software code. *See* Motoyama, 7:15–51. Motoyama teaches using revision numbers for managing objects (e.g., software code) stored in the project database. *See Id.* at 10:33–52. Appellant does not persuade us the Examiner errs in finding Motoyama's disclosure of a system that manages source code versions teaches the recited source code revision control system (SCRCS).

Appellant also argues the Examiner errs in relying on Wakui for teaching the "separate" aspect of the disputed limitation, because "Wakui does not appear to teach two separate systems." Appeal Br. 9. In particular, Appellant argues the Examiner errs in mapping Wakui's "parts server" and "parts lifetime managing server" to the SCRCS separate from the PMS, because "the parts server and parts lifetime managing server are a part of a same part lifetime management system." *Id.* at 10. This argument is unpersuasive. Wakui discloses its parts server and parts lifetime managing server as two separate computer systems connected over a network. *See* Wakui Fig. 3. Artisans of ordinary skill would have understood the general proposition that systems (e.g., Wakui's part lifetime management system) often include combinations of separate subsystems (e.g., a parts server separate from a part lifetime managing server). *Cf.* Ans. 11–12 (explaining that Motoyama teaches the functionality of the elements of the disputed limitation and "the Wakui reference merely establishes it is known to have these elements in separate locations").

Appellant further argues "Wakui does not appear to teach that a first system extracts a commit log from another system that is separate from the first system." Appeal Br. 9. This argument is unpersuasive. The rejection relies on Motoyama (i.e., not Wakui) for teaching extracting a commit log. *See* Final Act. 5–6, 16; *In re Keller*, 642 F.2d at 426 ("[O]ne cannot show non-obviousness by attacking references individually where, as here, the rejections are based on combinations of references.").

Appellant furthermore argues "Wakui is non-analogous art as it relates to solving a different problem than the current claims." Appeal Br. 9; Reply Br. 11–12. This argument is unpersuasive. A prior art reference is analogous to an application if it is either (i) in the same field of Appellant's endeavor, or (ii) reasonably pertinent to the particular problem with which the inventor is involved. *In re Clay*, 966 F.2d 656, 659 (Fed. Cir. 1992); *In re Bigio*, 381 F.3d 1320, 1325 (Fed. Cir. 2004). Wakui's disclosure automates coordination of information exchange (e.g., status) related to a design revision between a parts server and a parts lifetime managing server that are separate from each other (*see* Wakui ¶¶ 50–54, Fig. 3) is at least pertinent to the problem of automating coordination of task status between a SCRCS and a PMS that are separate from each other. We also agree in particular with the Examiner that, with respect to the limitations for SCRCS being separate from the PMS, Wakui pertinently "establishes it is known to have these elements in separate locations." *See* Ans. 12–13.

Claim 1 also recites

> changing the current task status data in the project management data store for the one of the plurality of tasks to reflect the new state of modifying the source code for the one of the plurality of tasks to synchronize the current task status data in the project management data store with the new state of modifying the source code as indicated in the commit log extracted from the [SCRCS].

The Examiner finds this limitation is taught by Motoyama's disclosure of updating status, e.g., from draft to official, for software code stored in a project database. *See* Final Act. 6–7; *see also* Ans. 13–14. Appellant contends Motoyama does not teach or suggest an SCRCS and therefore cannot teach or suggest this second disputed limitation. Appeal Br. 12–13; *see also* Reply Br. 13–14. This argument is unpersuasive because, as discussed *supra*, we agree with the Examiner that Motoyama's system for managing versions and status using a project database, including for software source code, teaches a source code revision control system (SCRCS), as recited.

Thus, we sustain the § 103 rejection of claim 1 and, along with it, claims 2, 4–6, 8, 9, 11–13, 15, 16, 18, 19.

Claim 3

Claim 3's parent claim 2 recites "searching the commit log for the keyword in view of a determination that the commit log is a new commit log." Appellant does not contest the Examiner's finding that Motoyama and Oddo teach the limitations recited by claim 2, including "a determination that the commit log is a new commit log." *See* Appeal Br. *passim*; Final Act. 18–20.

Claim 3 recites

> "[t]he method of claim 2, wherein determining whether the commit log is a new commit log comprises:"

> comparing a time stamp of the commit log to a time stamp of a commit log that was last searched; and
>
> searching the commit log for the keyword in view of a determination that the time of the time stamp of the commit log is after the time of the time stamp of the commit log that was last searched.

Appeal Br. 23 (Claims App'x).

The Examiner finds Guturu and Motoyama teach the specific method recited by claim 3 for performing claim 2's determination. Final Act. 56–59. Appellant contends the Examiner errs in relying on Guturu. Appeal Br. 13–14; Reply Br. 14–17. In particular, Appellant argues Guturu's "comparing a timestamp of a data record to a timestamp of a **data update request** is not a teaching of comparing a timestamp of a commit log to **a time stamp of a commit log that was last searched**." Reply Br. 16.

This argument is unpersuasive. The Examiner relies on Motoyama, not Guturu, for teaching the recited commit log. *See* Final Act. 5–6, 57–59. Appellant does not explain how or why the Examiner errs in relying on Guturu's time stamp teachings in combination with Motoyama's commit log teachings for the rejection of claim 3. *See* Appeal Br. 13–14; Reply Br. 14–17 (with respect to Guturu's teachings in combination with the other cited references, contending summarily only that "[f]urther, even when hypothetically combined with Motoyama, Best, Oddo, McKnight, Chigusa, and Wakui the combination does not teach or suggest determining whether a commit log is new by comparing a time stamp of the commit log to a time stamp of a commit log that was last searched, as recited by claim 3").

CONCLUSION

In summary:

Claim(s) Rejected	35 U.S.C. §	Reference(s)/Basis	Affirmed	Reversed
1–6, 8–13, 5–19	101	Eligibility	1–6, 8–13, 15–19	
1, 2, 4–6, 8, 9, 11–13, 15, 16, 18, 19	103(a)	Motoyama, Best, Oddo, McKnight, Chigusa, Wakui	1, 2, 4–6, 8, 9, 11–13, 15, 16, 18, 19	
3, 10, 17	103(a)	Motoyama, Best, Oddo, McKnight, Chigusa, Wakui, Guturu	3, 10, 17	
Overall Outcome			1–6, 8–13, 15–19	

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a)(1)(iv).

AFFIRMED