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.
13/486,806	06/01/2012	PAUL A. CROCKETT	GB920110035US1 8152-0191	6852

134531 7590 03/22/2018

Cuenot, Forsythe & Kim, LLC
20283 State Road 7, Ste. 300
Boca Raton, FL 33498

EXAMINER
AQUINO, WYNUEL S

ART UNIT	PAPER NUMBER
2199	

NOTIFICATION DATE	DELIVERY MODE
03/22/2018	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):

ibmptomail@iplawpro.com

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE PATENT TRIAL AND APPEAL BOARD

Ex parte PAUL A. CROCKETT

Appeal 2017-010913
Application 13/486,806
Technology Center 2100

Before MICHAEL J. STRAUSS, JEREMY J. CURCURI, and MICHAEL J. ENGLE, *Administrative Patent Judges*.

CURCURI, *Administrative Patent Judge*.

DECISION ON APPEAL

Appellant appeals under 35 U.S.C. § 134(a) from the Examiner's rejection of claims 1–20. Final Act. 1. We have jurisdiction under 35 U.S.C. § 6(b).

Claims 15–20 are rejected under 35 U.S.C. § 101 as directed to non-statutory subject matter. Final Act. 6.

Claims 1, 2, 4–9, 11–16, and 18–20 are rejected under pre-AIA 35 U.S.C. § 102(b) as anticipated by Conti (US 8,769,491 B1; July 1, 2014). Final Act. 7–13.

Claims 3, 10, and 17 are rejected under 35 U.S.C. § 103(a) as obvious over Conti and McAllister (US 2010/0205580 A1; Aug. 12, 2010). Final Act. 13–15.

We affirm-in-part.

STATEMENT OF THE CASE

Appellant's invention relates to "compiling object code for a secondary thread in a multi-threaded runtime environment in a computing system." Spec. ¶ 2. Claim 1 is illustrative and reproduced below, with the key disputed limitation emphasized:

> 1. A method of compiling source code into object code for a multi-threaded runtime environment, comprising:
>
> compiling source code into object code using a compilation engine;
>
> identifying marshalling attributes associated with method code intended for executing in a secondary thread; and
>
> *rewriting the marshalling attributes and the method code as marshaled method code for executing the method code in the secondary thread according to the identified marshalling attributes.*

ANALYSIS

THE NON-STATUTORY SUBJECT MATTER REJECTION OF CLAIMS 15–20

Contentions

The Examiner concludes claims 15–20 are directed to non-statutory subject matter because the recited (claim 15) "computer usable storage medium" includes signals. *See* Final Act. 6; *see also* Ans. 2.

Appellant argues the broadest reasonable interpretation of the claim language does not include a transitory signal. *See* App. Br. 9–20; *see also* Reply Br. 2–3.

Our Review

Appellant's Specification includes specific examples of computer usable storage media. Appellant's Specification discloses:

> Any combination of one or more computer readable medium(s) may be utilized. The computer readable medium may be a computer readable signal medium or a computer readable storage medium. A computer readable storage medium may be, for example, but not limited to, an electronic, magnetic, optical, electromagnetic, infrared, or semiconductor system, apparatus, or device, or any suitable combination of the foregoing. *More specific examples (a non-exhaustive list) of the computer readable storage medium would include the following: an electrical connection having one or more wires*, a portable computer diskette, a hard disk, a random access memory (RAM), a read-only memory (ROM), an erasable programmable read-only memory (EPROM or Flash memory), an optical fiber, a portable compact disc read-only memory (CD-ROM), an optical storage device, a magnetic storage device, or any suitable combination of the foregoing. In the context of this document, a computer readable storage medium may be any tangible medium that can contain (or store) a program for use by or in connection with an instruction execution system, apparatus, or device.

Spec. ¶ 19 (emphasis added).

Generally, the term "computer usable storage medium" encompasses transitory media and is, therefore, ineligible under § 101. *See Ex parte Mewherter*, 107 USPQ2d 1857, 1862 (PTAB 2013) (precedential) (holding recited machine-readable storage medium ineligible under § 101 because it encompassed transitory media in the absence of a specific definition or disclaimer). Furthermore, "physical but transitory forms of signal transmission such as . . . electrical signals through a wire, and light pulses through a fiber-optic cable" are not statutory subject matter. *In re Nuijten*, 500 F.3d 1346, 1353 (Fed. Cir. 2007). Here, Appellant's Specification does not expressly distinguish a computer usable *storage* medium from a *transitory* medium because Appellant describes an electrical connection having one or more wires (i.e., a medium for transitorily carrying a signal)

as an example of a computer usable storage medium. *See* Spec. ¶ 19. Thus, a computer usable storage medium may be wire, which may be a transitory medium, for example, a transmission line carrying a transitory signal. In light of the Specification, we conclude that the broadest reasonable interpretation of "computer usable storage medium" includes transitory forms.

We, therefore, sustain the Examiner's rejection of claims 15–20 under 35 U.S.C. § 101.

THE ANTICIPATION REJECTION OF CLAIMS 1, 2, 4–9, 11–16, AND 18–20 BY CONTI

Contentions

The Examiner finds Conti describes all limitations of claim 1. Final Act. 7–8. In particular, the Examiner finds Conti's annotations associated with code describe the key disputed limitation. Final Act. 7–8 (citing Conti, col. 4, ll. 15–49; Fig. 2A, elements 221, 222). More specifically, the Examiner finds Conti's compiled byte code describes the "marshaled method code" recited in claim 1. Final Act. 8.

Appellant presents the following principal argument: "The Examiner's analysis, however, fails to identify the claimed 'method code' and explain how this 'method code' and 'marshalling attributes' are rewritten." App. Br. 21; *see also* App. Br. 24–25 (citing Spec. Figs. 4 and 5) ("Appellant's [S]pecification recognizes a clear and unmistakable difference between 'rewriting' and 'compiling.').

In response, in pertinent part, the Examiner elaborates regarding the claim term "rewriting":

> the term "rewritten", under broadest reasonable interpretation, will be interpreted as parsing code, during run-time, into task portions (i.e. rewritten) executed in parallel according to attributes specifying the designated thread. Conti clearly teaches the equivalent of Applicant's argument above. Conti teaches: an interpreter that interprets (rewrites) code that allows for the determination of a particular thread to receive a task (method code) using annotations[.]

Ans. 17 (citing Conti, col. 5, ll. 20–33, col. 7, ll. 28–51, col. 16, ll. 6–15).

In the Reply Brief, Appellant further argues "[t]he claims require that the marshalling attributes are first identified and then rewritten." Reply Br. 4. Appellant further argues parsing is not rewriting. *See* Reply Br. 4–7.

Our Review

Conti discloses:

> Software developers that write code, such as source code, may add information to the code that affects the operation of the code. For example, a software developer may add annotations to source code in the form of comments. The annotations may be manually added to the source code by the developer or may be programmatically added via an annotation module. These annotations may not affect the working of the program but may be used to provide instructions to a user reviewing the source code. Alternatively, these annotations may provide instructions to an interpreter or compiler when the source code is interpreted or compiled, respectively. For example, in an exemplary embodiment, code that executes a task may be annotated with threading information. The threading information in the annotation may be examined by a dispatcher module at runtime to identify the thread to which the task should be dispatched. In contrast, existing mechanisms for executing scripts may not employ threading information in annotations of code for dispatching tasks to threads at runtime.
>
> Exemplary embodiments allow scripts to include references to separate code that executes a task, such as, but not

> limited to, non-scripting language code. An annotation associated with the separate code can provide threading information that may be accessed at runtime by a dispatcher module to determine which thread in the computing environment should execute the task. For example, a MATLAB-based script that is executed may make a call to a JAVA-based method. *The class definition for the JAVA-based method may include an annotation with threading information. The dispatcher module at runtime may either examine the class definition or compiled byte code to examine this threading information.* This dispatch mechanism allows a programmer writing or debugging scripts to utilize separate libraries of methods that are written in a different language while also exercising control over their manner of execution.

Conti, col. 4, ll. 15–49 (emphasis added).

Conti (col. 5, ll. 20–33) describes examination of an annotation and dispatching a task based on the annotation; the annotation may be compiled into a byte code representation, which is examined.

Conti (col. 7, ll. 28–51) further describes specifying threading policies in annotations; interpreting annotations may be delegated by the scripting language interpreter.

Conti (col. 16, ll. 6–15) describes parallel programming, in particular data parallel programs where data is parsed into multiple portions that are executed in parallel.

We agree with Appellant that claim 1 requires "identifying marshalling attributes associated with method code intended for executing in a secondary thread" and claim 1 further requires "rewriting the marshalling attributes and the method code as marshaled method code." That is, the marshalling attributes and method code are *first* identified, and *then* rewritten. For example, Appellant's Figure 9 depicts Stage 1 Source Code 32, Stage 2 Pre-Marshalled Object Code 212A, and Stage 3 Marshalled

Object Code 36. Notably, the Stage 2 actual code (and source equivalent code) includes the marshalling attributes and the method code, which are identified. Then, the marshalling attributes and the method code are re-written at Stage 3, as marshalled method code. The differences between Stage 2 code and Stage 3 code illustrate re-writing.

Turning to the cited disclosures in Conti, we do not agree with the Examiner that Conti's annotation 221 and code 222 in Figure 2A are *marshalled method code*. Rather, at best, Conti reasonably describes "marshalling attributes" (annotation 221 with threading information) and associated "method code" (code 222). Further, these are identified when "[t]he dispatcher module at runtime may either examine the class definition or compiled byte code to examine this threading information." Conti, col. 4, ll. 43–46. We recognize that Conti's annotation 221 and code 222 are similar to Appellant's Stage 2 Pre-Marshalled Object Code 212A in Figure 9. However, we do not see on the record presented by the Examiner where Conti's annotation 221 and code 222 are rewritten as *marshalled method code*.

Furthermore, if we were to accept the Examiner's finding that Conti's annotation 221 and code 222 in Figure 2A are *marshalled method code*, then we are unable to identify where Conti discloses the pre-marshalled code (prior to re-writing, "the marshalling attributes and the method code"). Lastly, the fact that compiled byte code may be parsed by an interpreter does not change our decision because the record does not adequately establish that Conti's annotation 221 and code 222 are marshalling attributes and method code that are rewritten as marshalled method code.

We, therefore, do not sustain the Examiner's rejection of claim 1. We also do not sustain the Examiner's rejection of claims 2 and 4–7, which depend from claim 1.

Independent claims 8 and 15 recite the same key disputed limitation. We, therefore, do not sustain the Examiner's rejection of claims 8 and 15. We also do not sustain the Examiner's rejection of claims 9, 11–14, 16, and 18–20, which variously depend from claims 8 and 15.

THE OBVIOUSNESS REJECTION OF CLAIMS 3, 10, AND 17 OVER CONTI AND MCALLISTER

The Examiner's application of McAllister fails to cure the above-discussed deficiency of Conti. *See* Final Act. 13–15.

We, therefore, do not sustain the Examiner's rejection of claims 3, 10, and 17.

DECISION

The Examiner's decision rejection claims 1–14 is reversed.

Because we sustain at least one ground of rejection for each of claims 15–20, the Examiner's decision rejecting claims 15–20 is affirmed.

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).

AFFIRMED-IN-PART