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.
11/618,214	12/29/2006	Kirk J. Krauss	CAM920060144US1_8132-0018	5454

73109 7590 11/21/2018

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

EXAMINER	
WU, JUNCHUN	

ART UNIT	PAPER NUMBER
2191	

NOTIFICATION DATE	DELIVERY MODE
11/21/2018	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 KIRK J. KRAUSS[1]

Appeal 2017-006267
Application 11/618,214
Technology Center 2100

Before ROBERT E. NAPPI, BRADLEY W. BAUMEISTER, and JESSICA C. KAISER, *Administrative Patent Judges*.

NAPPI, *Administrative Patent Judge*.

DECISION ON APPEAL

This is a decision on appeal under 35 U.S.C. § 134(a) of the Examiner's Final Rejection of claims 21 through 38, which constitute all the claims pending in this application.

We AFFIRM-IN-PART.

INVENTION

Appellant's invention relates to a method of runtime analysis for a computer program. *See* Abstract.

---

[1] According to Appellant, the real party in interest is IBM Corporation. App. Br. 1.

Claim 21 is illustrative of the invention and reproduced below:

> 21. A computer-implemented method for runtime analysis of an instrumented computer program, comprising:
>
> generating, for each of a plurality of separate memory regions used by the computer program during runtime, memory usage data associated with the computer program; and
>
> creating, automatically without user input from the memory usage data for each of the plurality of separate memory regions, a memory map; and
>
> presenting at least a portion of the memory map, wherein the memory map includes, based upon the memory usage data, a plurality of memory ranges of different types.

REJECTIONS AT ISSUE[2]

The Examiner has rejected claims 21 through 23, 26 through 29, 32 through 35, and 38 under pre-AIA 35 U.S.C. § 103(a) as being unpatentable over Goran Begic, *An Introduction to Runtime Analysis with Rational PurifyPlus*, The Rational Edge (2002) (hereinafter "Begic"), Spadari (US 2005/0102583 A1, May 12, 2005), D. Garbervetsky et al., *Program Instrumented and Run-Time Analysis of Scoped Memory in Java*, Electronic Notes in Theoretical Computer Science (2004) (hereinafter "Garbervetsky"), and Hilton (US 5,530,934, June 25, 1996). Answer 2–7.[3]

---

[2] Throughout this opinion we refer to the Appeal Brief, filed July 25, 2016 (hereinafter "App. Br."); the Reply Brief, filed February 2, 2017 (hereinafter "Reply Br."); Final Office Action mailed February 25, 2016 (hereinafter "Final Action"); and the Examiner's Answer, mailed on December 16, 2016 (hereinafter "Answer").

[3] We note the statement of the rejection does not include clams 27 through 38; however the Examiner identifies that these clams are rejected for the same reasons as the counterpart claims 21 through 26. Answer 9; Final

The Examiner has rejected claims 24, 30, and 36 under pre-AIA 35 U.S.C. § 103(a) as being unpatentable over Begic, Spadari, Garbervetsky, Hilton, and Jeff Campbell, *Memory Profiling for C/C++ with IBM Rationale Test RealTime and IBM Rational PurifyPlus Real Time* (2003) (hereinafter "Campbell"). Answer 7–8.

The Examiner has rejected claims 25, 31, and 37 under pre-AIA 35 U.S.C. § 103(a) as being unpatentable over Begic, Spadari, Garbervetsky, Hilton, and Karp et al. (US 2005/0283770 A1, Dec. 22, 2005) (hereinafter "Karp"). Answer 8–9.

ANALYSIS

We have reviewed the Examiner's rejections in light of Appellant's contentions that the Examiner has erred. Further, we have reviewed the Examiner's response to Appellant's arguments. We agree with Appellant's contention that the Examiner erred in rejecting dependent claims 23, 24, 29, 30, 35, and 36 under 35 U.S.C. § 103(a). However, Appellant's arguments have not persuaded us of error in the Examiner's decision to reject claims 21, 22, 25 through 28, 31 through 34, 37, and 38 under 35 U.S.C. § 103(a).

Independent claims 21, 27, and 33

Appellant argues that the Examiner's rejection of independent claims 21, 27, and 33 is in error. App. Br. 13–18; Reply Br. 2–6. These arguments

---

Action 10. We have identified all claims in the statement of the rejection, and note Appellant has argued the claims as being rejected in this manner. App. Br. 3–4.

present us with three issues with respect to representative claim 21: 1) whether the Examiner erred in finding the combination of the references teaches generating for separate memory regions, memory usage data for a (singular) program; 2) whether the Examiner erred in relying upon Garbervetsky's teaching of memory allocation when finding the combination of the references generates memory usage data; and 3) whether the Examiner erred in finding the combination of the references teaches a memory map includes a plurality of memory ranges of different types.

With respect to the first issue, Appellant argues the paragraph of Garbervetsky to which the Examiner cites (p. 116, section titled "Run-time analysis") discusses variable size regions of memory, but does not identify that these regions are to be used by one program or multiple programs. App. Br. 13–14; Reply Br. 2–3. Thus, Appellant argues the Examiner's rejection is in error as it does not teach the claim 21 limitation of "memory usage data associated with the computer program."

We are not persuaded of error by these arguments. The Examiner's rejection identifies that Begic teaches implementing a runtime analysis for a (singular) program. Answer 3. Further, the Examiner's rejection relies upon Garbervetsky to teach that separate memory regions are used during runtime. Answer 5. Thus, it is the combination of Begic and Garbervetsky that the Examiner uses to reject the disputed limitation, not Garbervetsky alone. Further, we additionally note that Garbervetsky, in section 5, which discusses simulating the behavior of different memory allocations to analyze fragmentation, discusses the simulation as being performed on a (singular) transformed program. Ans. 10 (citing Garbervetsky, 118-19). Thus, suggesting Garbervetsky's discussion of separate memory regions used

during runtime are applicable to a singular program. Accordingly, Appellant's arguments directed to the first issue have not persuaded us of error in the Examiner's rejection of representative claim 21.

With respect to the second issue, Appellant argues that Garbervetsky's discussion of separate memory regions being used during runtime is referring to memory regions that are allocated and not to memory regions that are used. App. Br. 15; Reply Br. 3–5. Further, Appellant argues:

> While Garbervetsky teaches analyzing the run-time behavior of allocation algorithms regarding memory fragmentation, this has nothing to do with memory usage. Instead, Garbervetsky involves memory allocation — i.e., the evaluation of algorithms that "manage a linked list of blocks where objects are allocated according to a first-fit or best-fit strategy."

App. Br. 16.

We are not persuaded of error by these arguments. As discussed above, the Examiner's rejection relies upon Garbervetsky to teach separate memory regions used during runtime in combination with the other references. We concur with Appellant that Garbervetsky's teachings involve allocation of memory, which is different from usage of memory. However, as discussed above, Garbervetsky's fragmentation analysis involves simulations of runtime using the different allocation, and as such, is also monitoring the usage of the memory.

Appellant's statement that this does not deal with memory usage is unsupported attorney argument. Garbervetsky states that the allocation can cause fragmentation, holes of temporarily unusable memory. *See* Garbervetsky, Sec. 4, Run Time Analysis. Thus, the simulation using the fragmentation analysis is considering how the memory is used.

Accordingly, Appellant's arguments have not persuaded us of error in the Examiner's rejection of representative claim 21.

With respect to the third issue, Appellant argues that claim 21 recites that the memory map includes a plurality of different types of memory and that Spadari does not teach this disputed limitation. App. Br. 17–18; Reply Br. 5–6. Appellant asserts that Spadari teaches a memory map with plural regions, but does not mention the types of the memory ranges. App. Br. 18. Further, Appellant asserts that the Examiner's finding that Spadari's ranges depicted in Figure 6, UART, MEM_1, MEM_2, and PERIPH_1, are different types is not supported by evidence. Reply Br. 6.

We are not persuaded of error by these arguments. Appellant's Specification does not define what constitutes a type of memory; rather, paragraph 89 provides a non-limiting list of what constitutes memory types. Spadari discusses that the data concerning memory usage can include things such peripheral registers (e.g., UART and PERIPH_1 in Fig. 6) and processor internal registers, i.e., different types. *See* Spadari para. 20. Also, Spadari teaches that the memory devices can include one or more of different types. *See* Spadari para. 22. Thus, there is ample evidence to support the Examiner's finding that Spadari teaches the memory map includes a plurality of different types of memory. As such, Appellant's arguments directed to the third issue have not persuaded us of error in the Examiner's rejection of representative claim 21. As Appellant's arguments only present these three issues with respect to representative claim 21, we sustain the Examiner's rejection of independent claim 21 and independent claims 27 and 33 grouped with claim 21.

Dependent claims 22, 28, and 34

Appellant argues the Examiner's rejection of claims 22, 28, and 34 is in error as the teachings of Begic cited by the Examiner do not teach the claimed feature of "intercepting . . . calls to memory management functions." App. Br. 19; Reply Br. 6–7. Specifically, Appellant argues that the cited teaching of Begic does not mention memory management functions, calls to memory management functions, or intercepting calls. App. Br. 19; Reply Br. 7.

We are not persuaded of error by these arguments. Appellant's arguments do not assert any particular interpretation of the claim terms memory management functions, calls to memory management function, or interrupts, to show error in the Examiner's rejection. Rather, Appellant merely asserts the portions of Begic cited by the Examiner refer to chains of calls and that this does not necessarily establish these calls are intercepted. Reply Br. 7.

To consider Appellant's arguments we look to Appellant's Specification to define the claimed "intercepting, by the runtime analysis code, calls to memory management functions." Appellant's Specification does not define these features, but rather states that "[t]hose familiar with runtime analysis tools, such as Rational® PurifyPlus™, for example, will appreciate that any of a variety of functions that relate to memory management can be intercepted and logged." Specification para. 84. As Begic's article is discussing uses of Rational PurifyPlus (*see* title), the Examiner's finding that the skilled artisan would recognize Begic teaches the claimed intercepting calls to memory management function is reasonable given Appellant's statement that people familiar with the Rational

PurifyPlus would appreciate that memory functions and calls can be tracked. Appellant only states the cited portion of Begic does not specifically mention the terms above, but Appellant does not persuasively argue why Begic fails to teach or suggest the disputed limitation, particularly in light of the statement about Rational PurifyPlus in Appellant's Specification discussed above. Accordingly, Appellant's arguments have not persuaded us of error in the Examiner's rejection of representative claim 22. Thus, we sustain the Examiner's rejection of claim 22 and claims 28 and 34 grouped with claim 22.

Dependent claims 23, 29, and 35

Appellant argues the Examiner's rejection of claims 23, 29, and 35 is in error as the Examiner has not shown that the combination of the references teaches the memory map has call flow information that indicates how the selected portion of memory was allocated. App. Br. 19–20; Reply Br. 7–8. Appellant argues that the portion of Spadari that the Examiner cites to teach this feature teaches presenting a portion of the memory map, but is silent as to an indication of how the memory was allocated. App. Br. 20; Reply Br. 8.

The Examiner, in response to Appellant's arguments, cites to paragraphs 32 and 34 of Spadari, which discuss traversing a memory map through a hierarchal structure. Answer 13. Further, the Examiner states "'the memory map may be traversed (e.g., by moving through a hierarchical structure providing increasing detail of the memory space) until a specific memory range or location of interest is found and selected' as call flow

information indicates how the selected portion of memory was allocated." Answer 13.

We are persuaded of error in the Examiner's rejection. We agree with the Examiner that Spadari teaches traversing a memory map. We do not agree with the Examiner's findings that the cited teachings include an indication of how the selected portion of the memory was allocated. Accordingly, we do not sustain the Examiner's rejection of claims 23, 29, and 35, all of which recite presenting the selected portion of the memory map call flow information indicating how the selected memory was allocated.

Dependent claims 26, 32, and 38

Appellant argues the Examiner's rejection of claims 26, 32, and 38 is in error as the Examiner has not shown that the combination of the references teaches presenting an expanded view indicating one component of the selected memory range, as recited in representative claim 26. App. Br. 21–22. Appellant argues that Spadari teaches selecting a memory range in an expanded view, but that the passage is silent with respect to a component in the memory range. App. Br. 22. Appellant asserts that paragraphs 94 and 95 of the Specification provide examples of components and that Spadari does not teach the claimed components. App. Br. 22; Reply Br. 9.

We are not persuaded of error by these arguments. The Examiner cites to Spadari as teaching this feature. Answer 13–14 (citing Spadari paras. 20, 33, and 36). We have reviewed the cited teachings of Spadari and concur with the Examiner. Appellant's Specification does not define

components, but provides a list of components in paragraph 94. The list includes, but is "not limited to, text, code, debug information or any of the other distinctive parts that may exist within a module." Thus, Appellant's Specification provides a broad list of things that constitute a component.

Spadari teaches the expanded view displays information related to the assembler instruction including program physical address, op-code, and assembler mnemonic. *See* Spadari para. 33. While Spadari's list of information in the display does not use the same names for the information as Appellant's list of information, Appellant's list is non-limiting and broadly includes "text." The information displayed in Spadari's system clearly includes text (*see, e.g.*, Figs. 6–10). Further, Spadari teaches the displayed information may include the op-code, which is a type of code, meeting another example of a component, as described in Appellant's Specification. Accordingly, Appellant's arguments directed to representative claim 26 have not persuaded us of error in the Examiner's rejection. Thus, we sustain the Examiner's rejection of claims 26, 32, and 38.

Dependent claims 24, 30, and 36

Appellant argues the Examiner's rejection of claims 24, 30, and 36 is in error as the Examiner has not shown that the combination of the references teaches selecting type of memory range as being a module, heap, or stack as recited in each of claims 24, 30, and 36. App. Br. 23–24; Reply Br. 10. Appellant argues that Campbell, the reference the Examiner relies upon for this limitation, teaches tracking memory allocated off a heap, but does not teach selecting a type of range. App. Br. 24.

The Examiner, in response to Appellant's arguments, finds that Spadari teaches memory ranges of different types and Campbell discloses searching the stack for data in an area between the bottom and top for pointers that fall within the heap allocated blocks, which the Examiner interprets as falling within heap blocks. Answer 14 (citing Campbell 4). Thus, the Examiner finds that the combination of the references teaches the disputed limitation.

Appellant's arguments have persuaded us of error in the Examiner's rejection. The cited portions of Spadari do not teach selecting a range based upon one of the claimed types. Campbell's teaching is of searching the stack for pointers and discloses that the pointers may fall within a heap, but the Examiner has not adequately explained how the reference teaches selecting the range based upon the memory type being a heap. Accordingly, we are persuaded of error in the Examiner's rejection of claims 24, 30, and 36, and we do not sustain the Examiner's rejection of these claims.

Dependent claims 25, 31, and 37

Appellant argues the Examiner's rejection of claims 25, 31, and 37 is in error as the Examiner has not shown that the combination of the references teaches indicating a boundary between two consecutive memory ranges of different types, as recited in representative claim 25. App. Br. 25; Reply Br. 12. Appellant argues that Karp teaches a boundary between two consecutive memory ranges, but does not teach that the boundary is either indicated or between memory ranges of different types. App. Br. 25; Reply Br. 12.

We are not persuaded of error by Appellant's arguments. The Examiner relies upon Karp as teaching indicating boundaries between memory ranges. Answer 15 (citing Karp para. 25 and Fig. 6). We concur with the Examiner's findings. Regarding Appellant's arguments that Karp does not teach indicating the boundary, we concur with the Examiner that Karp's Figure 6 shows an indication of the boundary. Also, as discussed above with respect to claim 21, Spadari teaches displaying memory ranges of different types, thus, also teaching the claimed feature that the boundary is between different types. Thus, we also find the references in combination teach the feature. Accordingly, Appellant's arguments have not persuaded us of error in the Examiner's rejection of representative claim 25. Thus, we sustain the Examiner's rejection of claim 25 and claims 31 and 37 grouped with claim 25.

DECISION

We reverse the Examiner's rejections of claims 23, 24, 29, 30, 35, and 36 under 35 U.S.C. § 103(a).

We affirm the Examiner's rejections of claims 21, 22, 25 through 28, 31 through 34, 37, and 38 under 35 U.S.C. § 103(a).

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). *See* 37 C.F.R. § 41.50(f).

AFFIRMED-IN-PART