I hereby certify that this correspondence is being filed using EFS-Web addressed to: Mail Stop Ex Parte Reexam, Central Reexamination Unit, Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450, on the date shown below.

Dated: April 15, 2011 Signature: /Robert T. Neufeld/
Atty. Reg. No. 48,394

Docket No. 13557.112021
(PATENT)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Reexamination of:

Yellin et al.

Patent No.: 6,061,520

Issue Date: May 9, 2000

For: METHOD AND SYSTEM FOR PERFORMING STATIC INITIALIZATION

Control No.: Not yet Assigned

Examiner: Not Yet Assigned

Art Unit: Not Yet Assigned

REQUEST FOR *EX PARTE* REEXAMINATION UNDER 37 C.F.R. § 1.510

Mail Stop Ex Parte Reexam
Attn: Central Reexamination Unit
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

King & Spalding, LLP (hereinafter, "Requester") submits, under the provisions of 37 C.F.R. § 1.510 *et seq.*, a Request for Reexamination (hereinafter, "Request") of claims 1-4 and 6-23 of U.S. Patent No. 6,061,520 (hereinafter "the '520 patent") entitled "Method and System for Performing Static Initialization," issued to Yellin et al. on May 9, 2000. The '520 patent is provided as Exhibit 1 to the Request.

In support of its request, Requester provides the following:

- The $2520.00 fee for requesting *ex parte* reexamination set forth in 37 C.F.R. § 1.20(c)(1) (37 C.F.R. § 1.510(a));

- A statement pointing out each substantial new question of patentability based on prior patents and printed publications (37 C.F.R. § 1.510(b)(1));

- An identification of every claim for which reexamination is requested, and a detailed explanation of the pertinency and manner of applying the cited prior art to every claim for which reexamination is requested (37 C.F.R. § 1.510(b)(2));

- A copy of every patent or printed publication relied upon or referred to in paragraphs (b)(1) and (b)(2) of 37 C.F.R. § 1.510, accompanied by an English language translation of all the necessary and pertinent parts of any non-English language patent or printed publication (37 C.F.R. § 1.510(b)(3));

- A copy of the entire patent including the front face, drawings, and specification/claims (in double column format) for which reexamination is requested, and a copy of any disclaimer, certificate of correction, or reexamination certificate issued in the patent. All copies must have each page plainly written on only one side of a sheet of paper ((37 C.F.R. § 1.510(b)(4)) (Exhibit 1); and

- A certification that a copy of the request has been served in its entirety on the patent owner at the address as provided for in 37 C.F.R. § 1.33(c). The name and address of the party served must be indicated. If service was not possible, a duplicate copy must be supplied to the Office ((37 C.F.R. § 1.510(b)(5)).

Pursuant to 35 U.S.C. § 303, the prior art references discussed in this Request raise "substantial new questions of patentability" with respect to claims 1-4 and 6-23 of the '520 patent.

TABLE OF CONTENTS

I. INTRODUCTION

Requester seeks reexamination of claims 1-4 and 6-23 of the '520 patent (Exhibit 1) under 35 U.S.C. §§ 302-307 and 37 C.F.R. § 1.510 *et seq*. The application for the '520 patent was filed on April 7, 1998. The '520 patent is assigned to Sun Microsystems, Inc.

This Request for Reexamination is related to the Request for Reexamination assigned Control No. 90/011,489, filed on Feb. 17, 2011. The U.S. Patent and Trademark Office ("USPTO") on March 23, 2011 granted that Request for Reexamination as to claims 1-4, 6-13, 15, 16, and 18-23 based on the Lewis reference as discussed in more detail below.

The claims of the '520 patent relate to a method and system for "initializing static arrays by reducing the amount of code executed by the virtual machine to statically initialize an array." *See* '520 patent, Abstract. Specifically, the Examiner allowed the '520 patent because its claims include the limitation of "play execution" of code, or, "simulating the code without execution." *See* Notice of Allowability (Jan. 4, 2000) (attached as Exhibit 2). However, this limitation was well-known prior to the filing of the application for the '520 patent: as described in further detail below, the Lewis reference discloses the execution of computer code in order to "maintain a running simulation of the MCode machine's stack." Further, Lewis discloses a platform-independent interpreter, *i.e.*, a virtual machine, and also discloses the simulation of the machine stack as an MCode instruction, and the insertion of that instruction as, among other possibilities, a constant.

In addition to the specific disclosures of Lewis, which anticipate the '520 patent, the Java Language Specification and the Java VM Specification provide a robust backbone and development guide for combination with and application of Lewis's simulated execution to the Java computer programming language. The Java computer programming language was developed in part by Frank Yellin and Richard D. Tuck, the two inventors named on the '520

patent. The Java Language Specification and Java VM Specification serve as prior art instructional guides on how to implement and optimize Yellin and Tuck's creation. Included are detailed descriptions for the implementation and optimization of specific concepts such as, *e.g.*, static initializers, arrays, storage into constant pools, and implementation on a virtual machine.

Further, the concept of simulated execution of byte codes against a memory without actual execution of the byte codes in order to identify, *e.g.*, a static initializer of an array, was commonly used in decompilers at and prior to the time of the invention. These decompilers, often favored by software pirates and hackers but also capable of assisting a developer to understand underlying code, do exactly what the '520 patent claims: they simulate the execution of compiled bytecode against a memory without actually executing the byte code in order to identify the underlying computer code. The prior art article by Todd A. Proebsting and Scott A. Watterson, "Krakatoa: Decompilation in Java (Does Bytecode Reveal Source?)," discusses the role and functionality of such decompilers, specifically focusing on the simulated execution prevalent in the Krakatoa decompiler. And the prior art article by Dave Dyer, "Java decompilers compared," is a review of three other prior art decompilers; the Dyer disclosure is especially relevant because it shows that the prior art decompilers were used to convert Java bytecode for static initialization of an array into a single source code instruction for performing the same operation.

The Requester has identified these five (5) printed publications that, alone or in combination, either anticipate or render obvious claims 1-4 and 6-23 of the '520 patent. The prior art printed publications pose a significant new question of patentability and they are not cumulative of information cited to or considered by the Examiner during prosecution of the '520 patent. These printed publications anticipate or render obvious each element of the '520 patent,

including the purportedly novel application of the "play execution" of computer code that was the basis for the '520 patent's allowability.

Accordingly, at least in view of these listed prior art references and the substantial new questions of patentability that they raise, the Requester respectfully requests the issuance of an order for reexamination, and further requests that claims 1-4 and 6-23 be canceled. The Requester respectfully requests that this Request be afforded special dispatch in accordance with 35 U.S.C. § 305 and 37 C.F.R. § 1.550.

The Requester further respectfully requests that the Director provide an order of action dates to accompany the decision ordering reexamination of the '520 patent.

II. STATEMENT UNDER 37 C.F.R. § 1.510 (B)(1) POINTING OUT SUBSTANTIAL NEW QUESTIONS OF PATENTABILITY

The new prior art references not previously considered by the Examiner raise substantial new questions with respect to the patentability of claims 1-4 and 6-23 of the '520 patent. Section II.A provides an overview of the '520 patent. Section II.B summarizes certain aspects of the law regarding reexamination. Section II.C summarizes the evidentiary standards applicable to reexamination. Section II.D provides a list of all prior art relied upon in this Request. Section II.E provides a list of other supporting documents discussed in this Request. Section II.F provides a summary of pending litigation involving the '520 patent. Section II.G provides an identification of the substantial new questions of patentability raised in this Request. Section II.H provides an overview of the substantial new questions of patentability raised in this Request.

A. Overview of the '520 Patent

The '520 patent broadly claims a method of receiving code, simulating, or "play executing," the execution of computer code without actually running the code in order to identify the target operation, and then creating an instruction or shortcut allowing the processing

component to perform the target operation. This method is the basis of claim 6, the broadest of the claims of the '520 patent. In a preferred embodiment, the '520 patent seeks to improve "conventional systems for initialization of static arrays by reducing the amount of code executed by the virtual machine to statically initialize an array." *See* '520 patent, Abstract; *see also* '520 patent, Claim 1. Claims 1, 6, 12, and 18 are the independent claims.

Claim 1 recites:

> A method in a data processing system for statically initializing an array, comprising the steps of:
>
> compiling source code containing the array with static values to generate a class file with a clinit method containing byte codes to statically initialize the array to the static values;
>
> receiving the class file into a preloader;
>
> simulating execution of the byte codes of the clinit method against a memory without executing the byte codes to identify the static initialization of the array by the preloader;
>
> storing into an output file an instruction requesting the static initialization of the array; and
>
> interpreting the instruction by a virtual machine to perform the static initialization of the array.

Claim 6 recites:

> A method in a data processing system, comprising the steps of:
>
> receiving code to be run on a processing component to perform an operation;
>
> play executing the code without running the code on the processing component to identify the operation if the code were run by the processing component; and
>
> creating an instruction for the processing component to perform the operation.

Claim 12 recites:

> A data processing system comprising:
>
> a storage device containing:
>
> a program with source code that statically initializes a data structure; and
>
> class files, wherein one of the class files contains a clinit method that statically initializes the data structure;
>
> a memory containing:
>
> a compiler for compiling the program and generating the class files; and

> a preloader for consolidating the class files, for play executing the clinit method to determine the static initialization the clinit method performs, and for creating an instruction to perform the static initialization; and
>
> a processor for running the compiler and the preloader.

And Claim 18 recites:

> A computer readable medium containing instructions for controlling a data processing system to perform a method, comprising the steps of:
>
> receiving code to be run on a processing component to perform an operation;
>
> simulating execution of the code without running the code on the processing component to identify the operation if the code were run by the processing component; and
>
> creating an instruction for the processing component to perform the operation.

The '520 patent issued from U.S. Patent Application Serial No. 09/055,947. The '520 patent does not claim priority to any previous documents. During prosecution of the underlying application for the '520 patent, the applicants received one Office Action, rejecting Claims 1 and 3 under 35 U.S.C. § 102(b) as anticipated by Cierniak, "Briki: an optimizing Java compiler," objecting to Claims 2, 4, and 5 as depending from a rejected claim, and allowing Claims 6-23. Interestingly, the Examiner allowed claim 6, the broadest claim of the '520 patent, but rejected claim 1, which is merely a preferred embodiment of the broader claim 6. *See* Non-Final Rejection (July 21, 1999) at 3 (attached as Exhibit 3). The Examiner noted that claim 6 (and the other claims deemed allowable) were distinct from the prior art at least because they disclosed "explicit operations (e.g., initialization, allocation, manipulation and simulation) with respect to 'play' execution." *See id.* at 4. In response, the Applicant filed an Amendment, modifying claim 1 to include the following underlined language: "<u>simulating execution of</u> [play executing] the byte codes of the clinit method against a memory <u>without executing the byte codes</u> to identify the static initialization of the array by the preloader." *See* Amendment (Oct. 19, 1999) at 2 (attached

as Exhibit 4). The Examiner then allowed the application on January 4, 2000. *See* Notice of Allowability. The Examiner declared that:

> the cited prior art, either singly or in combination, fails to anticipate or render obvious the simulation of execution with respect to class initialization, without executing byte codes. Cierniak, "Briki: An Optimizing Java Compiler," teaches of comparing the execution of byte codes, but fails to address the issue of simulating the process without execution.

See id. at 2. Thus, the Examiner's statements and the claim amendments show that the step of simulating the execution of code without actually executing that code, *i.e.*, "play executing" the code, was the purportedly novel element of the application leading to its allowability.

B. Aspects of the law governing reexamination

1. Citation of prior art

Any person at any time may file a request for reexamination by the Office of any claim of any patent on the basis of any prior art cited under the provisions of section 301." 35 U.S.C. § 302. Section 301 limits prior art to "patents or printed publications." 35 U.S.C. § 301.

MPEP 2128 classifies a reference as a printed publication if it is accessible to the public:

> A reference is proven to be a 'printed publication' 'upon a satisfactory showing that such *document* has been disseminated or otherwise made available to the extent that persons interested and ordinarily skilled in the subject matter or art, exercising reasonable diligence, can locate it.'

In re Wyer, 655 F.2d 221, 210 USPQ 790 (C.C.P.A. 1981) (quoting *I.C.E. Corp. v. Armco Steel Corp.*, 250 F. Supp. 738, 743, 148 USPQ 537, 540 (S.D.N.Y. 1966)).

2. "Old" prior art can raise a significant new question of patentability

The fact that a prior art reference was cited or even previously considered by an examiner does not preclude use of that reference to find a substantial new question of patentability. *See* 35 U.S.C. § 303(a); MPEP Section 2258.01; *see also In re Swanson*, 540 F.3d 1368, 1380-81 (Fed. Cir. 2008) (holding that consideration of a prior art reference in previous litigation and in

an original examination does not preclude a finding of a SNQ based on the same prior art reference in reexamination).

A combination of such "old art" and art newly cited during the reexamination proceeding may raise a SNQ. *See* MPEP Section 2258.01. The Patent Office may even find a SNQ based exclusively on previously cited references.

> For example, a SNQ may be based solely on old art where the old art is being presented/viewed in a new light, or in a different way, as compared with its use in the earlier concluded examination(s), in view of a material new argument or interpretation presented in the request.

See id.

3. Obviousness standard under *KSR*

The Supreme Court recently relaxed the Federal Circuit's requirement of a "teaching/suggestion/motivation test," and instead held that "[t]he combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results." *KSR Int'l Co. v. Teleflex Inc. et al.*, 550 U.S. 398, 416 (2007). The Court noted that "[w]hen a work is available in one field of endeavor, design incentives and other market forces can prompt variations of it, either in the same field or a different one. If a person of ordinary skill can implement a predictable variation" of an existing system, then "§103(a) likely bars its patentability." *Id.* at 417. *KSR* also held that "if a technique has been used to improve one device, and a person of ordinary skill in the art would recognize that it would improve similar devices in the same way, using the technique is obvious" if within his or her skill. *See id.*

On October 10, 2007, after the prosecution of the '520 patent had come to a close, the USPTO released Examination Guidelines for Determining Obviousness Under 35 U.S.C. 103(a)

in View of the Supreme Court Decision in *KSR Int'l Co. v. Teleflex Inc.*, 72 Fed. Reg. 195 at 57526 (the "PTO Guidelines"). The PTO Guidelines adopt the rationales from the *KSR* decision for determining obviousness. One of the rationales is "'Obvious to Try' – Choosing from a Finite Number of Identified, Predictable Solutions, With a Reasonable Expectation of Success." To reject a claim on this basis, the PTO Guidelines note that pertinent factors to consider are whether "there had been a finite number of identified, predictable potential solutions to the recognized need or problem," and "one of ordinary skill in the art could have pursued the known potential solutions with a reasonable expectation of success." *Id.* at 57532. The PTO Guidelines have been incorporated into the MPEP's examination guidelines for determining obviousness under 35 U.S.C. § 103. *See* MPEP 2141.

Additionally, the Federal Circuit has applied the *KSR* obviousness standard to combine multiple embodiments disclosed in a single prior art reference. *Boston Sci. Scimed, Inc. v. Cordis Corp.*, No. 2008-1073, 2009 U.S. App. LEXIS 588, at *24 (Fed. Cir. Jan. 15, 2009) (holding that a person of ordinary skill would have been motivated to combine one embodiment found in a patent reference with a second, separate embodiment found in the same patent reference).

4. Prior art references need not be enabling in an obviousness inquiry

Moreover, prior art references need not be enabling in the context of an obviousness inquiry. As stated in the MPEP:

35 U.S.C. 103(a) REJECTIONS AND USE OF INOPERATIVE PRIOR ART

> "Even if a reference discloses an inoperative device, it is prior art for all that it teaches." *Beckman Instruments v. LKB Produkter AB*, 892 F.2d 1547, 1551, 13 USPQ2d 1301, 1304 (Fed. Cir. 1989). Therefore, "a non-enabling reference may qualify as prior art for the purpose of determining obviousness under 35 U.S.C.

> 103." *Symbol Techs. Inc. v. Opticon Inc.*, 935 F.2d 1569, 1578, 19 USPQ2d 1241, 1247 (Fed. Cir. 1991).

MPEP 2121.01; *see also* MPEP 2145; *Amgen Inc. v. Hoechst Marion Roussel, Inc.*, 314 F.3d 1313, 1357 (Fed. Cir. 2003) (holding that under 35 U.S.C. § 103, "a reference need not be enabled; it qualifies as prior art, regardless, for whatever is disclosed therein.") (citations to other cases omitted).

5. Claims of the patent are to be broadly construed

In a reexamination proceeding, claims are to be given their broadest construction consistent with the specification. *See In re Icon Health & Fitness, Inc.*, 496 F.3d 1374, 1379 (Fed. Cir. 2007) ("During reexamination, as with original examination, the PTO must give claims their broadest reasonable construction consistent with the specification.").

C. Evidentiary standards

If the prior references raise a substantial question of patentability of at least one claim of the patent, then a substantial new question of patentability is present. *See* MPEP 2242. A prior art patent or printed publication raises a substantial question of patentability where there is a substantial likelihood that a reasonable examiner would consider the prior art patent or printed publication important in deciding whether or not the claim is patentable. *Id.*

D. Prior art references relied upon in this Request

In accordance with 37 C.F.R. § 1.510, reexamination of claims 1-4 and 6-23 of the '520 patent is requested in view of the prior art publications listed below, which raise substantial new questions of patentability. This Request will demonstrate how claims 1-4 and 6-23 of the '520 patent are anticipated or rendered obvious in view of the following prior art references:

1. Brian T. Lewis, L. Peter Deutsch, and Theodore C. Goldstein. *Clarity MCode: A Retargetable Intermediate Representation for Compilation*, ACM, IR '95, 1/95, San Francisco, California, USA (1995) (hereinafter "Lewis"), provided as Exhibit 5.

2. James Gosling, Bill Joy, & Guy Steele. *The Java™ Language Specification*, Addison-Wesley (1st ed. 1996) (hereinafter the "Java Language Specification"), provided as Exhibit 6.

3. Sun Microsystems Computer Corp. *The Java™ Virtual Machine Specification*, Release 1.0 Beta DRAFT, (Aug. 21, 1995) (hereinafter the "Java VM Specification"), provided as Exhibit 7.

4. Dave Dyer, *Java Decompilers Compared*, JavaWorld.com (July 1, 1997) (hereinafter "Dyer"), provided as Exhibit 8.

5. Todd A. Proebsting and Scott A. Watterson. *Krakatoa: Decompilation in Java (Does Bytecode Reveal Source?)*, Proceedings of the Third USENIX Conference on Object-Oriented Technologies and Systems, Portland, Oregon (June 1997) (hereinafter "Proebsting"), provided as Exhibit 9.

E. Supporting documents discussed in this Request

The following documents are provided to assist the Examiner in understanding the Request, including claim charts and references providing background information:

1. Claim Chart based on Lewis, provided as Exhibit 10.

2. Claim Chart based on Lewis, the Java Language Specification, and the Java VM Specification, provided as Exhibit 11.

3. Claim Chart based on Lewis, Dyer, and Proebsting, provided as Exhibit 12.

F. Current Litigation

The Requester is aware of at least one current litigation matter involving the '520 patent. On August 12, 2010, Oracle America, Inc. filed a complaint in the U.S. District Court for the Northern District of California alleging that Google, Inc. infringed the '520 patent. The case is styled *Oracle America, Inc. v. Google, Inc.*, Civil Action No.: 3:10-cv-03561 WHA. A Joint Case Management Statement for the case, dated November 18, 2010, provides for a claim construction hearing in the case to take place on April 20, 2011. Non-expert discovery will end on July 29, 2011. The deadline for filing dispositive motions is September 8, 2011.

G. Identification of Substantial New Questions of Patentability

In this Request, substantial new questions of patentability for claims 1-4 and 6-23 of the '520 patent are identified in accordance with 37 CFR § 1.510(b)(1) as follows:

1. Anticipation under 35 U.S.C. § 102(b) based on the Lewis reference.

    a. Claims 14 and 17 are unpatentable under 35 U.S.C. § 102(b) as being anticipated by Lewis.

2. Obviousness under 35 U.S.C. § 103(a) based on the Lewis, Java Language Specification, and Java VM Specification references.

    a. Claims 1-4 and 6-23 are unpatentable under 35 U.S.C. § 103(a) as being obvious over Lewis in view of the Java Language Specification and further in view of the Java VM Specification.

3. Obviousness under 35 U.S.C. § 103(a) based on the Lewis, Dyer and Proebsting references.

    a. Claims 1-4 and 6-23 are unpatentable under 35 U.S.C. § 103(a) as being obvious over Lewis in view of Dyer and further in view of Proebsting.

H. Overview of Substantial New Questions of Patentability

As discussed above, claim 6 of the '520 patent broadly claims a method of receiving code, simulating the execution of (or "play executing") that code without actually running the code in order to identify the target operation, and then creating an instruction or shortcut allowing the processing component to perform the target operation. And claim 1 generally applies this method to the static initialization of arrays, with the specification of the '520 patent focusing largely on applying the method of Claim 1 to the <clinit> method to perform static initialization.

As discussed above, during prosecution, the patentability of claims 1 and 6 hinged on the method step of execution of computer code without actually executing that code. But this execution method was publicly known, available, and practiced at the time of the invention of

the '520 patent. In fact, the Lewis reference is a Sun Microsystems printed publication disclosing the very same "play execution" that Sun Microsystems would later attempt to patent in the underlying application for the '520 patent, which was filed over three years after the publication of the Lewis article. The USPTO recently issued an Order Granting Ex Parte Reexamination based on the Lewis reference as to claims 1-4, 6-13, 15, 16, and 18-23. The application of the Lewis reference to claims 14 and 17 is described in more detail below and in the attached claim chart. Also, and in addition to the application of the Lewis reference to claims 14 and 17, this Request outlines below (and more specifically in the attached claim charts) the combination of the Lewis disclosure with the well-known provisions of the Java Language Specification and Java VM Specification and also with the well-known decompiler technology as disclosed by Dyer and Proebsting, rendering obvious claims 1-4 and 6-23 of the '520 patent.

Lewis reference

Lewis discloses a "high-level, machine-independent intermediate representation [called] *MCode* (for "middle code"). *See* Lewis at 119. Since the Lewis reference was published in January of 1995 it is prior art to the '520 patent under 35 U.S.C. § 102(b), given a priority date for the '520 patent of April 7, 1998. The Lewis reference was not in front of the Patent Office during the prosecution of the application that matured into the '520 patent nor is it cumulative to the prior art considered by the Patent Office during the prosecution of the '520 patent.

Lewis is a publication discussing Sun Microsystems' development of a dialect of the C++ computer programming language, labeled "*Clarity C++*." *See* Lewis at 119. The Lewis disclosure is primarily directed to the discussion of a high-level, machine-independent intermediate code representation tool for support in the compilation of the Clarity language. *See id.* Lewis refers to this representation of code as *MCode*. *See id.* The Lewis code generator play

executes computer code in order to "maintain a running simulation of the MCode machine's stack." *See id.* at 126. Thus, Lewis discloses simulating execution of code to generate MCode instructions, including instructions for statically initializing an array. More specifically, and as related to claim 14, Lewis discloses an interpreter that is platform-independent, *i.e.*, a virtual machine. *See* Lewis at 127. The platform-independent virtual machine disclosed by Lewis is capable of, and indeed is directed towards, the implementation of an instruction which is created from simulated code: "MCode calls to other MCode procedures are implemented using SPARC instructions and execute the procedure's machine language entry code." *Lewis* at 127.

As can be seen from Figure 1 of the Lewis reference, reproduced below, the MCode generator receives code that is configured to be run on a processing component to perform an operation. *See* Lewis at 121.

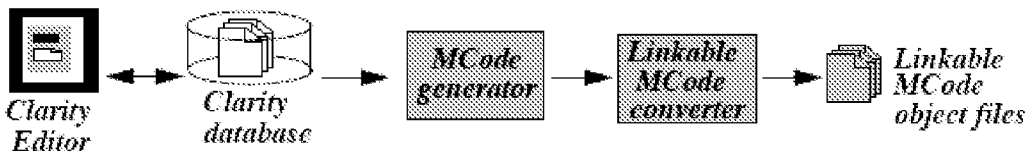

Figure 1: The development-time portion of the Clarity MCode system

As disclosed in Figure 1, the components of the Lewis system "receive" code to be run on a processing component. For example, the "MCode generator" block in Fig. 1 receives Clarity C++ code. This Clarity C++ code necessarily results in the performance of an operation.

The MCode developed by Lewis seeks to "reduce program memory requirements." *See* Lewis at 122. MCode reduces the program memory requirements by creating a "pickle," i.e. "a compact, platform independent encoding of the MCode information into a sequence of bytes. This pickle can later be internalized or unpickled to reconstruct the original MCode. The MCode

for each procedure is pickled separately to support procedure-at-a-time processing." *See* Lewis at 125.

The Lewis reference simulates execution of the code without actually running the code in order to identify the targeted output of a given section of code. "The code generator 'executes' MCode instructions in order to maintain a running simulation of the MCode machine's stack." *See* Lewis at 126. Once the targeted output of a given section of code is known, a shortcut, referred to as a "CGValue," is created, which represents the state of the individual entries of the simulated stack. "These entries include constants, variable references, previously 'executed' subexpressions, and procedure or method calls." *See id.* These CGValues operate as a set of shortcut instructions, such that "[g]ood code can be generated" when "the value of the expression is needed." *See id.* In other words, an instruction is created (the CGValue) as a constant and stored for later reference; such storage of constants inherently discloses entry into a constant pool.

More specifically, and as related to claim 17, Lewis discloses the entry of a constant as a "CGValue." "The second C++ base class, CGValue, describes values during compilation. The code generator 'executes' MCode instructions in order to maintain a running simulation of the MCode machine's stack. Concrete subclasses of CGValue represent the state of the individual entries on the simulated stack. These entries include constants, variable references, previously 'executed' subexpressions, and procedure or method calls." *See* Lewis at 126. Here, Lewis implements a CGValue, which represents a simulation of the machine's stack. *See id.* This can include an array: "MCode's types currently include integer, real, pointer, array, procedure, bit field, struct, union, interface, implementation, and void." *See id.* at 122. Thus Lewis anticipates the disclosure of claim 17 of the '520 patent, the created instruction includes an entry into a

constant pool. Requester further notes that a SNQ was found in the related reexamination (Control No. 90/011,489) with respect to claim 2 which is similar to claim 17.

A reasonable examiner would have considered the teachings of the Lewis reference to be important in determining whether or not claims 14 and 17 of the '520 patent were patentable. As detailed in the claim chart in Exhibit 10, the Lewis reference anticipates claims 14 and 17 of the '520 patent. For this reason, the Lewis reference raises a substantial new question of patentability with respect to claims 14 and 17 of the '520 patent.

Lewis in view of Java Language Specification and the Java VM Specification

Lewis, as described above, is prior art to the '520 patent under 35 U.S.C. § 102(b), given a priority date for the '520 patent of April 7, 1998, and was not before Patent Office during the prosecution of the application that matured into the '520 patent nor is it cumulative to the prior art considered by the Patent Office during the prosecution of the '520 patent. The Java Language Specification, First Edition, was published in 1996, and is thus also prior art to the '520 patent under 35 U.S.C. § 102(b). And the Java VM Specification was published on August 21, 1995, and is therefore prior art to the '520 patent under 35 U.S.C. § 102(b).

The Java VM Specification and twenty-one of the 800+ pages of the Java Language Specification are listed as being considered by the Examiner, but the combination of the Lewis reference in view of the entire Java Language Specification and the Java VM Specification was not in front of the Patent Office during the prosecution of the application that matured into the '520 patent nor is the combination cumulative to the prior art considered by the Patent Office during the prosecution of the '520 patent.

The Lewis reference, as disclosed above, clearly recites the "play execution" that was the key reason for the patentability of the '520 patent. One of ordinary skill in the art would have

looked to apply the simulated, or "play" execution of Lewis to the predominant computer languages in widespread use, including the ubiquitous Java computer programming language. The motivation to combine the Lewis reference with the Java Language Specification and the Java VM Specification is reinforced by the fact that Peter Deutsch, a co-author of the Lewis reference, also participated in reviewing drafts of the Java Language Specification. *See* Java Language Specification at xxiv.

The Java computer programming language was developed in part by Frank Yellin and Richard D. Tuck, the two inventors named on the '520 patent. *See* Java Language Specification at xxiii ("The final form of the language was defined by James Gosling, Bill Joy, Guy Steele, **Richard Tuck**, **Frank Yellin**, and Arthur van Hoff, with help from Graham Hamilton, Tim Lindholm and many other friends and colleagues.") (emphasis added). The Java computer programming language, like the C++ language referenced in Lewis, is a class-based, object-oriented programming language. *See id.*; *see also* Lewis at 119 ("The Clarity C++ programming language is a dialect of C++ being developed in Sun Microsystems Laboratories."). A key functionality of the Java computer programming language is its ability to allow developers to write a program one time and then be able to run it anywhere on the Internet, through the use of virtual machines. *See* Java Language Specification at xxiii & 1.

The Java Language Specification is a prior art instructional guide on how to implement and optimize Yellin and Tuck's creation. The Java Language Specification teaches all aspects of the '520 patent save the play execution taught by Lewis. For example, the Java Language Specification teaches static initialization, *see, e.g.*, Java Language Specification at 128, the initialization of arrays, *see, e.g., id.*, the utilization of virtual machines, *see, e.g., id.* at 201-202, the initialization of static fields from constants, *see, e.g.*, *id.* at 221, allocation of a stack, *see,*

e.g., *id.* at 336, the compilation to and reading of byte codes, *see, e.g.*, *id.* at 1, and every other limitation of the '520 patent. Exhibit 11 provides exemplary disclosures of the abundant and detailed disclosures of these limitations within the Java Language Specification. Further, the Java VM Specification is an instruction guide, taking the developer through a step-by-step guide of the Java virtual machine structure for the implementation of the Java Language tool kit described in the Java Language Specification.

It would have been obvious to one of ordinary skill in the art at the time of the invention to take the play execution taught by Sun Microsystems' Lewis publication and apply it to Sun Microsystems' well-known Java computer programming language, using the Java Language Specification and the Java VM Specification as instructional guides. By taking this natural step, the artisan would be in possession of all aspects of the technology later claimed by the application that led to the '520 patent.

A reasonable examiner would have considered the teachings of the Lewis reference in view of the Java Language Specification and the Java VM Specification to be important in determining whether or not claims 1-4 and 6-23 of the '520 patent were patentable. As detailed in the claim chart in Exhibit 11, Lewis in view of Java Language Specification and further in view of Java VM Specification renders obvious claims 1-4 and 6-23 of the '520 patent. For this reason, the Lewis reference in view of the Java Language Specification and further in view of the Java VM Specification combination raises a substantial new question of patentability with respect to claims 1-4 and 6-23 of the '520 patent.

Lewis in view of Dyer and Proebsting

Lewis, as described above, is prior art to the '520 patent under 35 U.S.C. § 102(b), given a priority date for the '520 patent of April 7, 1998. Dyer was published on July 1, 1997, and is

thus prior art to the '520 patent under 35 U.S.C. § 102(a). And Proebsting was published in June of 1997, and is thus also prior art to the '520 patent under 35 U.S.C. § 102(a). Neither Lewis, nor Dyer, nor Proebsting was in front of the Patent Office during the prosecution of the application that matured into the '520 patent, nor is the combination of Lewis, Dyer, and Proebsting cumulative of the prior art considered by the Patent Office during the prosecution of the '520 patent.

Lewis, as discussed above, is a publication discussing Sun Microsystems' development of a dialect of the C++ computer programming language, labeled "Clarity C++." See Lewis at 119. The Lewis code generator play executes computer code in order to "maintain a running simulation of the MCode machine's stack." See id. at 126.

This play execution was well-known in the prior art, as evidenced by the prevalence of various decompilers. Proebsting provides a detailed disclosure of one of these decompilers, and the means by which a decompiler is able to take compiled byte code and deconstruct that code to identify the underlying source code. For example, Proebsting discloses "decompiling Java bytecode into Java source." *See* Proebsting at Abstract. This decompilation is possible because the decompiler performs a "[s]ymbolic execution of the bytecode [to] create[] the corresponding Java source expressions." *See id.* at § 2. Even more specifically, this decompilation method works because "[s]ymbolic execution simulates the Java Virtual Machine's evaluation stack with strings that represent the source-level expressions being compounded." *Id.* This disclosure provides a clear and fully operable decompiler system (the Krakatoa decompiler system) that play executes computer code against a memory without executing the byte codes to identify the underlying code, which may inherently include static initializers and the static initialization of an array.

Along with the Krakatoa decompiler disclosed by Proebsting, numerous other decompilers were available in the prior art. The Dyer reference is a prior art article disclosing not only a review of three popular decompilers, but also the results of their decompilation, in the form of actual underlying computer code. As part of Dyer's review of the DejaVu, Mocha, and WingDis decompilers, the Dyer reference describes identifying the static initialization of an array. For example, Dyer discloses that the WingDis compiler, "[w]hen decompiling this same static initializer . . . produced equally beautiful and syntactically correct code." *See* Dyer at 3. With respect to the "explicit operations" discussed by the Examiner, *see* Non-Final Rejection at 4, Dyer discloses sections of code which allow for "static initializer(s)" and the manipulation of code using "inline code inside constructors," *see* Dyer at 3. Thus, the Dyer reference discloses explicit operations (e.g., static initializers) with respect to "play" execution (such as that disclosed in Proebsting).

The fact that some of Dyer's decompiled code may be considered "illegal" should not detract from the fact that the decompilation of static array initialization byte code into a single expression or instruction was a well-known technique at the time the '520 patent was filed. In fact, the '520 patent discloses creating a constant pool entry that is not a standard Java virtual machine construct. Thus, one of ordinary skill in the art would have considered this contant pool construct (i.e., CONSTANT_Array (*see* '520 patent at 8:54-9:13)) as an "illegal" Java constant pool entry. Regardless, Dyer discloses that the WingDis decompiler produced "syntactically correct code." *Dyer* at 3. That is, it was not "illegal" code. Accordingly, Dyer shows that a person of ordinary skill would arrive at an operable embodiment of the '520 patent claims by combining the well-known decompilation techniques of Dyer and Proebsting with the Lewis system.

A reasonable examiner would have considered the teachings of Lewis in view of Dyer and further in view of Proebsting to be important in determining whether or not the claims of the '520 patent were patentable. As detailed in the claim chart in Exhibit 12, Lewis in combination with Dyer and Proebsting renders obvious claims 1-4 and 6-23 of the '520 patent. For this reason, the Lewis reference in view of Dyer and further in view of Proebsting raises a substantial new question of patentability with respect to claims 1-4 and 6-23 of the '520 patent.

III. DETAILED EXPLANATION UNDER 37 C.F.R. § 1.510(B)(2) OF THE PERTINENCY AND MANNER OF APPLYING THE CITED PRIOR ART TO EVERY CLAIM FOR WHICH REEXAMINATION IS REQUESTED

The detailed explanation herein under 37 C.F.R. § 1.510(b)(2) comprises a summary of the reasons for unpatentability of the claims (set forth below) supported by detailed Claim Charts. This detailed explanation describes the pertinence and manner of applying the prior art references to the claims of the '520 patent.

A. Rejections of Claims

1. Claims 14 and 17 are unpatentable under 35 U.S.C. § 102(b) as being anticipated by the Lewis reference.

The Lewis reference was published in January of 1995. The printed publication is prior art to the '520 patent under 35 U.S.C. § 102(b). The Lewis reference was not in front of the Patent Office during the prosecution of the application that matured into the '520 patent nor is it cumulative of the prior art considered by the Patent Office during the prosecution of the '520 patent. As set forth in detail in the Claim Chart attached as Exhibit 10, the Lewis reference discloses each of the elements of claims 14 and 17 of the '520 patent.

2. Claims 1-4 and 6-23 are unpatentable under 35 U.S.C. § 103(a) as rendered obvious by Lewis in view of the Java Language Specification and further in view of the Java VM Specification.

The Lewis reference was published in January of 1995. The printed publication is prior art to the '520 patent under 35 U.S.C. § 102(b). The Java Language Specification, First Edition, was published in 1996, and is thus also prior art to the '520 patent under 35 U.S.C. § 102(b). The Java VM Specification was published on August 21, 1995, and is therefore prior art to the '520 patent under 35 U.S.C. § 102(b). The combination of the Lewis reference, the Java Language Specification and the Java VM Specification was not in front of the Patent Office during the prosecution of the application that matured into the '520 patent, nor is it cumulative of the prior art considered by the Patent Office during the prosecution of the '520 patent. As set forth in detail in the Claim Chart attached as Exhibit 11, Lewis in view of the Java Language Specification and the Java VM Specification renders obvious each of the elements of claims 1-4 and 6-23 of the '520 patent.

3. **Claims 1-4 and 6-23 are unpatentable under 35 U.S.C. § 103(a) as rendered obvious by Lewis in view of Dyer and further in view of Proebsting.**

The Lewis reference was published in January of 1995. The printed publication is prior art to the '520 patent under 35 U.S.C. § 102(b). The Dyer reference was published on July 1, 1997. It is prior art to the '520 patent under 35 U.S.C. § 102(a). And the Proebsting reference was published in June of 1997. It is prior art to the '520 patent under 35 U.S.C. § 102(a). The combination of Lewis in view of Dyer and further in view of Proebsting was not in front of the Patent Office during the prosecution of the application that matured into the '520 patent, nor is the combination cumulative to the prior art considered by the Patent Office during the prosecution of the '520 patent. As set forth in detail in the Claim Chart attached as Exhibit 12, Lewis in view of Dyer and further in view of Proebsting renders obvious each of the elements of claims 1-4 and 6-23 of the '520 patent.

IV. CONCLUSION

For the reasons provided herein, Requester respectfully submits that the prior art submitted herewith raises substantial new questions of patentability as to claims 1-4 and 6-23 of the '520 patent because, as discussed above, claims 1-4 and 6-23 of the '520 patent are either anticipated or rendered obvious in view of the prior art publications discussed herein. Accordingly, reexamination of claims 1-4 and 6-23 of the '520 patent is respectfully requested, finally rejecting these claims.

The undersigned further notes the standards set forth at 37 C.F.R. 1.550(f) wherein the reexamination Requester will be sent copies of Office actions issued during the reexamination proceedings as well as served (by the patent owner) with any document filed in the reexamination proceeding in accordance with 37 C.F.R. 1.248. (*See* MPEP §§ 2264 and 2266.)

If the Patent Office determines that a fee and/or other relief is required, Requester petitions for any required relief including authorizing the Commissioner to charge the cost of such petitions and/or other fees due in connection with the filing of this document to **Deposit Account No. 11-0980** referencing Docket No. 13557.112021.

As identified in the attached Certificate of Service and in accordance with 37 C.F.R. §§ 1.33(c) and 1.510(b)(5), a copy of the present request is being served to the address of the attorney or agent of record.

April 15, 2011

Respectfully submitted,

By Robert T. Neufeld
Patent Attorney
Registration No. 48,394
KING & SPALDING LLP
1180 Peachtree Street
Atlanta, Georgia 30309