IN THE UNITED STATES DISTRICT COURT
FOR THE DISTRICT OF DELAWARE

ORACLE CORPORATION and ORACLE U.S.A., INC.,	)	
Plaintiffs-Counterdefendants,	)	
v.	)	C.A. No. 06-414 (SLR)
EPICREALM LICENSING, LP,	)	
Defendant-Counterclaimant.	)	

ORACLE'S OPPOSITION TO EPICREALM'S MOTION FOR PARTIAL SUMMARY JUDGMENT OF LITERAL INFRINGEMENT

MORRIS, NICHOLS, ARSHT & TUNNELL LLP
Mary B. Graham (#2256)
James W. Parrett, Jr. (#4292)
1201 N. Market Street
P.O. Box 1347
Wilmington, DE 19899-1347
(302) 658-9200
mgraham@mnat.com
jparrett@mnat.com
Attorneys for Oracle Corporation and Oracle U.S.A., Inc.

OF COUNSEL:

James G. Gilliland
Theodore T. Herhold
Joseph A. Greco
Robert J. Artuz
Eric M. Hutchins
TOWNSEND AND TOWNSEND AND CREW LLP
379 Lytton Avenue
Palo Alto, CA 94301
(650) 326-2400

Dorian Daley
Peggy E. Bruggman
Matthew M. Sarboraria
ORACLE CORPORATION
ORACLE U.S.A., INC.
500 Oracle Parkway
Redwood Shores, CA 94065

Dated: August 21, 2008

TABLE OF CONTENTS

Page

I. NATURE AND STAGE OF THE PROCEEDINGS 1

II. SUMMARY OF ARGUMENT 1

III. STATEMENT OF FACTS 4

A. The Requirements of the '554 Patent 4

1. Intercepting the Dynamic Web Page Request at the Web Server 7

2. Routing the Dynamic Web Page Request from the Web Server to a Dispatcher 9

3. Dispatching the Dynamic Web Page Request to a Page Server 10

4. Page Server Releasing the Web Server 11

B. Oracle's Accused Products and EpicRealm's Shifting Infringement Theories 11

1. Web Cache 12

2. IAS 13

3. Database 15

4. Summary Chart of EpicRealm's Shifting Infringement Theories 16

C. EpicRealm's Specious "Dynamic Load Balancing" Arguments 16

IV. ARGUMENT 18

A. The Accused Oracle Products Do Not Infringe Claim 11 Because They Lack The Central Limitations Of The Claim 18

B. Dynamic Web Page Requests Are Not "Intercepted" Or Diverted At Either Of The Alleged Web Servers (OHS Or Web Cache) 19

1. EpicRealm's theories against IAS and Database fail because requests are never intercepted or diverted from OHS, the alleged Web server 19

2. EpicRealm's theory against Web Cache fails because requests are never intercepted or diverted from Web Cache, the alleged Web server 22

TABLE OF CONTENTS(con't)

Page

C. The Accused Oracle Products Do Not Have A Dispatcher That Performs Dispatching....24

1. EpicRealm's unsupported assertion that mod_oc4j is the "dispatcher" of IAS fails because mod_oc4j is merely a function of the Web server, OHS....24

2. EpicRealm's Database theory fails because Oracle Database lacks a dispatcher and does not perform dispatching....25

a) EpicRealm's theory that OCI and JDBC are dispatchers fails because they are merely part of the Web server, OHS....25

b) EpicRealm's new theory, if allowed, that OC4J, using JDBC, is the "dispatcher" of Oracle Database fails because OC4J is a "page server"....27

c) Oracle Database does not perform dispatching because neither alleged dispatcher analyzes or examines a Web page request to make an informed selection of a page server....28

3. EpicRealm's theory that Web Cache is both the "Web server" and the "dispatcher" fails....30

D. The Accused Oracle Products Cannot Perform The Releasing Limitation....31

1. EpicRealm's theories against IAS and Database fail because OHS, the accused Web server, is never released by an alleged "page server"....31

a) EpicRealm's "implicit" releasing theory fails....31

b) EpicRealm's "explicit" releasing theory fails....34

(1) OC4J, the alleged page server, does not send the TCP ACK message; the operating system sends ACK messages....34

(2) Even if the operating system is part of the "page server," the sending of any ACK message does not release OHS to process other requests....36

TABLE OF CONTENTS(con't)

Page

(3) Even if "freeing of resources" hypothetically could satisfy the releasing limitation, epicRealm's theory fails 37

2. EpicRealm's theory against Web Cache fails because Web Cache, the accused Web server, is never released by an alleged "page server".....38

a) EpicRealm's "implicit" theory fails.....38

b) EpicRealm's "explicit" theory fails 38

E. Oracle Does Not Make Or Sell A Machine Readable Medium That Meets The Limitations Of Claim 11.....39

V. CONCLUSION.....40

TABLE OF AUTHORITIES

Page

Cases

Amgen Inc. v. Hoechst Marion Roussel, Inc.
314 F.3d 1313 (Fed. Cir. 2003) 35

Bayer AG v. Elan Pharm. Research Corp.
212 F.3d 1241 (Fed. Cir. 2000) 18

Dynacore Holdings Corp. v. U.S. Philips Corp.
363 F.3d 1263 (Fed. Cir. 2004) 19

Elan Pharma Int'l Ltd., v. Abraxis Bioscience, Inc.
C.A. No. 06-438 GMS (D. Del. Dec. 6, 2007) 27

Inline Connection Corp. v. AOL Time Warner Inc.
472 F. Supp. 2d 604 (D. Del. 2007)..... 27

Padcom, Inc. v. Netmotion Wireless, Inc.
No. Civ.03-983-SLR, 2006 WL 416865 (D. Del. Feb. 22, 2006) 39

TIP Sys., LLC v. Phillips & Brooks/Gladwin, Inc.
529 F.3d 1364 (Fed. Cir. 2008) 21, 25, 32

Vitronics Corp. v. Conceptronic, Inc.
90 F.3d 1576 (Fed. Cir. 1996) 35

Plaintiffs Oracle Corporation and Oracle USA, Inc. (jointly "Oracle") submit this brief in opposition to defendant epicRealm's Motion for Partial Summary Judgment of Literal Infringement.

I. NATURE AND STAGE OF THE PROCEEDINGS

Oracle incorporates by reference its summary of the Nature and Stage of Proceedings set forth in its Opening Claim Construction Brief ("Oracle CC Br."), D.I. 203, filed on July 31, 2008.

II. SUMMARY OF ARGUMENT

1. EpicRealm moves for partial summary judgment of literal infringement *only* on claim 11 of the '554 patent. As shown in Oracle's Motion for Summary Judgment of Noninfringement, the same claim limitations that are lacking in claim 11, a "machine readable medium" claim, also are lacking in the other asserted "system" and "method" claims of the patents-in suit.

2. EpicRealm's conclusory characterizations of Oracle's products are insufficient for a proper infringement analysis. EpicRealm's brief and the conclusory expert declaration of Dr. Finkel are devoid of reference to the Oracle source code. Dr. Finkel reviewed the source code for only one minor option of Oracle's products ("metrics-based load balancing"). He failed to review the source code relevant to Oracle's noninfringement defenses discussed in this brief. The source code, however, conclusively establishes the structures and functions of Oracle's products. Dr. Clark, Oracle's expert, undertook a detailed examination of the relevant source code and explains in his declaration why Oracle's products do not meet the claim limitations. A proper analysis of the Oracle source code shows conclusively that Oracle's products do not infringe claim 11 of the '554 patent, the only claim on which epicRealm moves, or any other asserted claim.

3. EpicRealm's brief does not accurately portray the patents-in-suit. The patents-in-suit claim a specific "partitioned architecture" for discriminating between "dynamic Web page requests" and other requests, and processing the dynamic requests differently. Web page requests for dynamic content must be identified and "intercepted" at the Web server so that the Web server does not continue to process them. The intercepted dynamic requests must then be "routed" from the Web server to a software program called a "dispatcher." The dispatcher software program must "analyze" or "examine" the requests and dispatch them to a "page server," which must in turn "release" the Web server to process other requests.

4. EpicRealm's infringement arguments gloss over these unambiguous claim limitations, and effectively read these limitations out of the claims, because Oracle products do not meet them. By ignoring specific claim limitations, epicRealm simply labels the structures and functions of Oracle's products in whichever way epicRealm finds expedient to suit its various infringement theories. Thus, epicRealm alleges that the *same* parts of Oracle's products which perform the *same* functions satisfy *different* claim limitations in epicRealm's different and conflicting infringement theories. The alleged "Web server" in one theory morphs into the alleged "page server" in another theory; the "page server" of the first theory includes the "dispatcher" in yet another theory, but that theory alleges that the "page server" is now the "data source" of the first theory; and so on.

5. EpicRealm's theory against Oracle's Internet Application Server ("IAS") product fails because:

- Requests for dynamic Web pages are not "intercepted" at the alleged Web server, Oracle HTTP Server ("OHS"). OHS processes *all* requests, regardless of whether they are for dynamic or static content, in the same way.

- There is no "dispatcher." EpicRealm's attempt to label OHS, a single software program, as both the "Web server" and the "dispatcher" fails because the Web

server program must "route" requests *to* a "dispatcher" program. OHS cannot route requests to itself.

- The alleged "page server," Oracle Containers for Java ("OC4J"), does not "release" OHS to process other requests.

6. EpicRealm's theory against Oracle's Database product fails because:

- Requests for dynamic Web pages are not "intercepted" at the alleged Web server, OHS. OHS processes *all* requests, regardless of whether they are for dynamic or static content, in the same way.

- There is no "dispatcher." EpicRealm labels as "dispatchers" two Application Programming Interfaces ("APIs"), Java Database Connectivity ("JDBC") and Oracle Call Interface ("OCI"). Neither of these APIs is a software program to which OHS can "route" requests.

- In addition, neither JDBC or OCI "analyzes" or "examines" requests for Web pages, which the patent establishes are Uniform Resource Locators ("URLs"), and thus neither alleged "dispatcher" performs "dispatching."

- The alleged page server, a Real Application Cluster ("RAC") instance of the Database, does not "release" OHS to process other requests.

7. EpicRealm's attempt to add a new theory against Database should not be allowed because it was not disclosed in its responses to interrogatories or in its expert's report or deposition. Even if allowed, however, this new theory fails for the reasons stated above and because:

- There still is no "dispatcher." OC4J and JDBC do not jointly constitute a dispatcher, because when requests are routed to OC4J, OC4J is the *only* page generating software ("page server"), not a RAC Database instance. If OC4J were the dispatcher or part of the dispatcher, there would be no page server.

- There is still no "dispatching." The "information" epicRealm now alleges that JDBC or OCI "examines" or "analyzes" is not a "dynamic Web page request" because it is not a URL.

8. EpicRealm's theory against Oracle's Web Cache product fails because:

- Requests for dynamic Web pages are not "intercepted" at the alleged Web server, Web Cache. Web Cache processes *all* requests, regardless of whether they are for dynamic or static content, in the same way.

- There is no "dispatcher." EpicRealm's attempt to label Web Cache a single software program, as both the "Web server" and the "dispatcher" fails because the Web server program must "route" requests *to* a "dispatcher" program. Web Cache cannot route requests to itself.

- The alleged "page server," IAS (apparently some combination of OHS and OC4J) does not "release" Web Cache to process other requests.

9. Instead of focusing on what the patents-in-suit do claim, epicRealm tries to read into the claims a requirement that the general term "dispatching" include a type of dynamic "load balancing" of page servers, which appears nowhere in claim 11 or in any of the other asserted claims. EpicRealm neither invented dynamic load balancing, nor included it as a requirement of any of the asserted claims. Oracle's products have dynamic load balancing capabilities, though customers rarely, if ever, use them. The presence of this rarely used option is irrelevant to Oracle's noninfringement defenses. EpicRealm argues speciously that: Oracle believes using dynamic load balancing is "critical" to its products; Oracle's market share increased by adding dynamic load balancing to its products; and Oracle decided to use dynamic load balancing based on the disclosures of the patents-in-suit. These baseless arguments epitomize the contrived nature of epicRealm's case.

III. STATEMENT OF FACTS

A. The Requirements of the '554 Patent

The '554 patent, as its title notes, claims a "system for managing *dynamic* Web page generation requests." A1 (emphasis added).[1] The Summary of the Invention provides that:

[1] References to evidence are to the Appendix of Exhibits in Support of Oracle's Claim Construction Brief and Motions Filed on July 31, 2008 (D.I. 214 and 215), and to the Supplemental Appendix of Oracle Exhibits Filed on August 21, 2008. The Appendix contains Exhibits A1-A71; The Supplemental Appendix begins with Exhibit A72.

"Specifically, the present invention claims a method and apparatus for managing *dynamic* Web page generation requests." Col. 2:17-19 (emphasis added). The patent distinguishes between a "static" Web page request and a "dynamic" Web page request. The patent, discussing the prior art, states:

> The scope and ability of a Web page designer to change the content of the Web page was limited by the static nature of the Web pages. Once created, a Web page remained static until it was manually modified. This in turn limited the ability of Web site managers to effectively manage their Web sites.

Col. 1:41-46. The patent goes on to describe dynamic Web page generation requests:

> The Common Gateway Interface (CGI) standard was developed to resolve the problem of allowing dynamic content to be included in Web pages. CGI "calls" or procedures enable applications to generate dynamically created HTML output, thus creating Web pages with dynamic content. Once created, these CGI applications do not have to be modified in order to retrieve "new" or dynamic data. Instead, when the Web page is invoked, CGI "calls" or procedures are used to dynamically retrieve the necessary data and to generate a Web page.

Col. 1:47-56 (emphasis added). As a result of this prior art development of allowing dynamic content to be included in Web pages, Web site "administrators no longer have to constantly update static Web pages." Col. 1:58-59.

The '554 patent thus claims a system for distinguishing requests for Web pages with dynamic content from other requests, such as static Web page requests, and managing the dynamic Web page requests differently. Consequently, it is critical to the patented invention that the Web server make a distinction between dynamic and static web page requests so that dynamic requests are handled differently. To manage the requests for dynamic content, the '554 patent proposes "utilizing a partitioned architecture to facilitate the creation and management of custom Web sites and servers." Col. 4:51-53. The partitioned architecture is shown in Figure 4 of the patent:

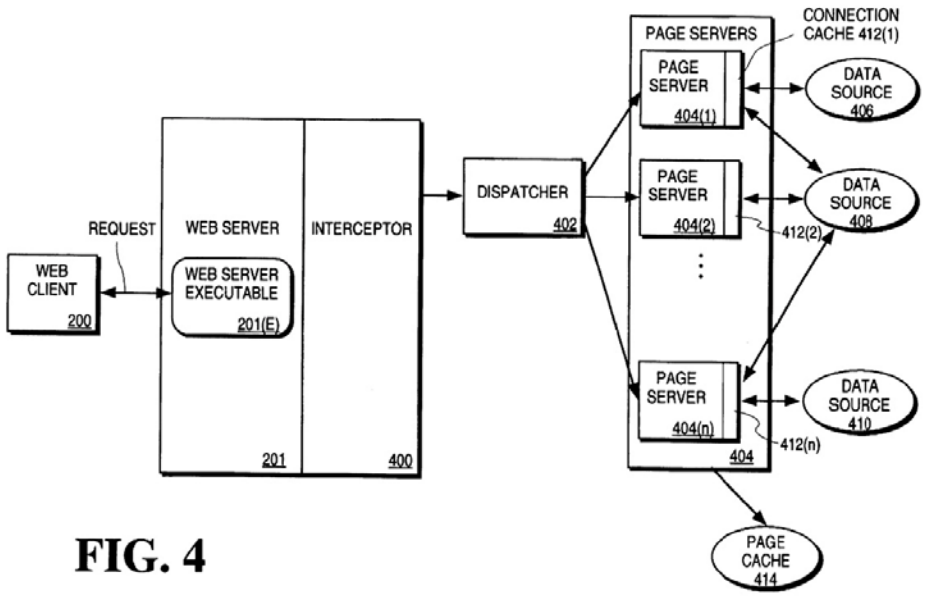


FIG. 4

As can be seen, the partitioned architecture has distinct parts that are used to manage the requests for dynamic content: a Web server including a Web server program (executable), an interceptor,[2] a dispatcher program, a page server program, and a data source. The patent provides that the Web server program (executable) may be on a different machine than the dispatcher program and page server program, or may be on the same machine. An "executable" is another name for a software "program." An example of a familiar software program is Microsoft Word. It is a single "executable" program file called "Word.exe." The Word program has different features and functions contained within it—e.g., print, preview, open file—but these are all part of the same program or "executable." Declaration of Dr. Paul C. Clark in Opposition to EpicRealm's Motion for Partial Summary Judgment of Literal Infringement ("Clark Opp. Decl.") ¶ 50.

[2] Claim 11 requires "intercepting at the Web server" but does not require an "interceptor." This is true of the other asserted claims, except claim 10 (which requires both an "interceptor" and "intercepting").

If the Web server program, dispatcher program, and page server program are placed on the same machine, therefore, they still are separate programs because they must be able to "route" requests to each other. These requirements are specifically set forth in claim 11, which claims a "machine readable medium" that has the code sufficient to create such a system:

> A machine readable medium having stored thereon data representing sequences of instructions, which when executed by a computer system, cause said computer system to perform the steps of:
>
> *routing* a dynamic Web page generation request from a *Web server* to a *page server*, said *page server* receiving said request and *releasing* said *Web server* to process other requests wherein said routing step further includes the steps of *intercepting said request at said Web server*, *routing* said request from said *Web server* to a *dispatcher*, and *dispatching* said request to said *page server*;
>
> processing said request, said processing being performed by said page server while said Web server concurrently processes said other requests; and
>
> dynamically generating a Web page, said Web page including data retrieved from one or more data sources.

A1 (emphasis added). The first recited step in the claim refers to routing the dynamic Web page generation request from a Web server to a page server. The claim language is unambiguous that, in operation, the steps to accomplish this routing begin with "intercepting said request at said Web server." The subsequent required steps are: the intercepted dynamic Web page request must be "routed" from the Web server to a "dispatcher" and the "dispatcher" must "dispatch" the request to a "page server." The page server must then perform four distinct acts: (1) "receive" the request; (2) "release" the Web server to process other requests; (3) "process" the request; and (4) dynamically generate a Web page. These claim requirements will be discussed in turn below.

1. Intercepting the Dynamic Web Page Request at the Web Server

In the claimed invention, requests for dynamic content are identified and "intercepted at the Web server." The Web server program (executable) does not process these dynamic Web

page requests. With reference to Figure 4, the patent states:

> Web client 200 issues a URL request that is processed to determined proper routing. In this embodiment, the request is routed to Web server 201. *Instead of Web server executable 201(E) processing the URL request*, however, Interceptor 400 intercepts the request and routes it to Dispatcher 402.

Col. 4:55-60 (emphasis added). EpicRealm's description of the intercepting requirement is based on an "example" of the Web server processing static requests and *not* processing dynamic requests:

> Not all requests are handled by the Web server, according to the patent. For a request that will not be processed at the Web server, "Interceptor 400 intercepts the request and routes it to the Dispatcher 402." For example, if the request is for a static Web page, *i.e.*, one that is relatively easy to locate and retrieve, the request may be processed by the Web server itself—in other words, the static page is located by the Web server and then sent back to the Web client. If, however, the request is for a dynamic Web page, the Web page to be sent back may be created by one of a number of "page servers" . . . .

ER Br. at 5 (citations omitted). EpicRealm's use of the term "may" attempts to leave open the possibility that (1) static Web page requests could be processed by some part of the system other than the Web server, and (2) dynamic Web page requests could be processed by the Web server.

There is no indication in the patent's specification or claims, however, that "static Web page requests" will be sent to a page server for processing. To the contrary, the claimed "interception" of requests for dynamic content at the Web server must include identifying those requests, as opposed to static requests, and diverting them from the Web server. Similarly, there is no indication in the specification or claims that "dynamic Web page requests" will be processed by the Web server. The only disclosure in the specification and claims is that requests for dynamic content will be "intercepted at the Web server" and processed by a "page server."

EpicRealm argues that "[w]hen the request is received at the Web server, the Web server initially has the duty to process the request." ER Br. at 5. Nevertheless, both the specification

and the claims require that the Web server *not* have the "duty" to complete processing requests for dynamic content, but that these requests be "intercepted" and processed by a "page server." Indeed, this is the very purpose of the claimed invention: Off-loading the processing of dynamic web page requests to separate "page servers" so that the Web server itself does not have to process them. Oracle's products do not distinguish between requests for static content and requests for dynamic content. *See* Section IV.B, *infra*.

2. Routing the Dynamic Web Page Request from the Web Server to a Dispatcher

Claim 11 requires "routing" the intercepted request for a dynamic Web page "from said Web server to a dispatcher." The dispatcher of the patent is a separate software program to which the Web server program routes requests. Oracle addressed this issue in its Opening Claim Construction Brief, at 27-30, and will only briefly summarize that discussion here. Figure 4 accurately illustrates the dispatcher as separate from the Web server and Web server executable. The patent provides that it is most advantageous for the dispatcher to be on a separate machine from the Web server, further demonstrating that the dispatcher is a separate software program. Col. 5:8-19. The patent provides that the dispatcher may be placed on the same machine as the Web server, but that the dispatcher could be off-loaded onto a separate machine as a Web site grows, which is only possible if it is a separate software program from the Web server program. Col. 5:20-36; Clark Opp. Decl. ¶¶ 46-51. The "routing" requirement of the claims cannot be

satisfied unless the dispatcher is a separate program because the Web server cannot "route" requests to itself. *Id.*[3]

EpicRealm glosses over the need for a dispatcher software program separate from the Web server program, contending that a claim construction of "dispatcher" is unnecessary. EpicRealm seeks to avoid the "dispatcher" limitation because Oracle's products do not have any "dispatcher" separate from its alleged Web server software. *See* Section IV.C, *infra*.

3. Dispatching the Dynamic Web Page Request to a Page Server

The parties agree that to "dispatch" a dynamic Web page request to a page server, the dispatcher must "examine" or "analyze" the request to make an informed selection of which page server should process the request. *See* Oracle CC Br. at 31 n.21. As the patent makes clear in its description of Fig. 4, the dynamic Web page request is the URL (Uniform Resource Locator) request that comes from the Web client (*e.g.*, a Web browser such as Internet Explorer):

> Web client 200 issues a *URL request* that is processed to determined proper routing. In this embodiment, the request is routed to Web server 201. Instead of Web server executable 201(E) processing the *URL request*, however, Interceptor 400 intercepts the request and routes it to Dispatcher 402.

Col. 4:55-60 (emphasis added). EpicRealm's new infringement theory against the Oracle Database is predicated on the "Web page request" being something other than the URL. *See* Section IV.C.2(c), *infra*.

---

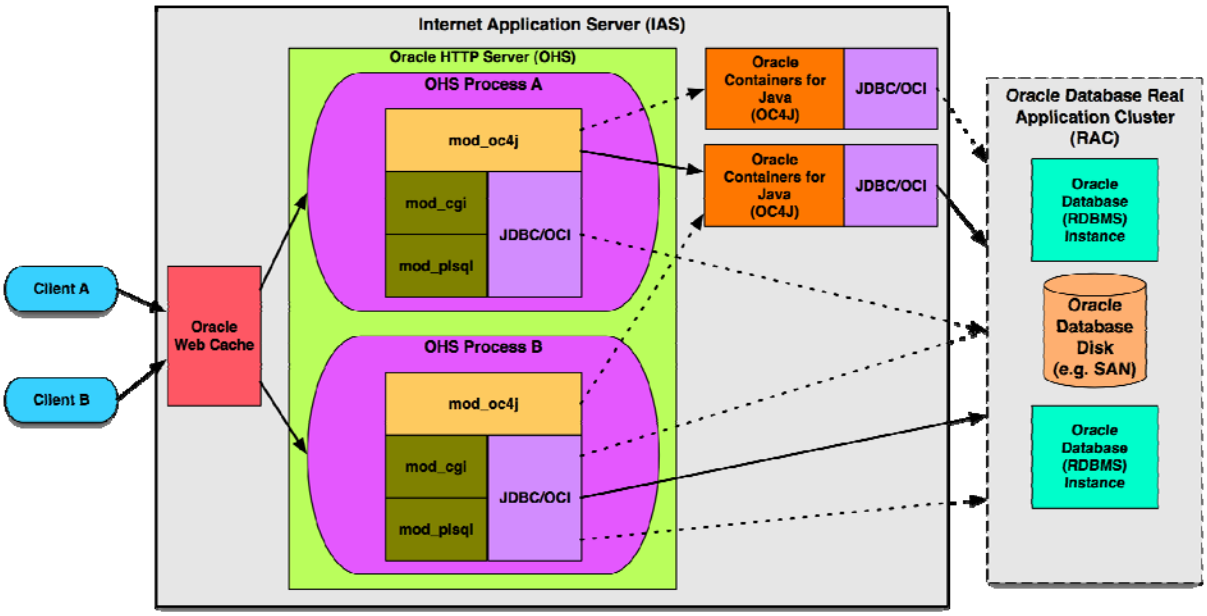
[3] The Texas court recently revisited and upheld its construction of the "releasing" requirement of the claims ("said page server receiving said request and said page server performing an act (separate from merely receiving the request) to free the Web server to process other requests." A88 (8/7/08 Order). The court noted that the Web server and the page server must be separate programs running on separate operating systems because if they are not, "the Web server would effectively be 'routing' requests from itself to itself." *Id.* at 7 n.3. The same problem would exist if the Web server is not a separate program from the dispatcher.

4. Page Server Releasing the Web Server

EpicRealm's description of the patent seeks to read the "releasing" limitation out of the claims. As discussed above, epicRealm mischaracterizes the role of the claimed Web server as having a "duty" to process dynamic Web page requests, which the Web server never, in fact, has in the claimed invention. EpicRealm argues that the page server "assumes the processing duties for that request" and "[i]n this way, the Web server is released to perform other tasks." ER Br. at 5-6. As discussed above with respect to claim 11, however, the page server must perform at least four distinct acts: (1) "receive" the dynamic Web page request; (2) "release" the Web server to process other requests; (3) "process" the dynamic Web page request; and (4) dynamically generate a Web page. The alleged page server's "processing" of a request is thus an act required by the claims. If, by performing that act, the page server *ipso facto* "releases" the Web server to process other requests, then the separate "releasing" limitation would be rendered superfluous. *See, e.g.*, Oracle CC Br. at 18-20.

B. Oracle's Accused Products and EpicRealm's Shifting Infringement Theories

Oracle sells three products—Web Cache, Internet Application Server ("IAS") and Database—which epicRealm mixes and matches to create three different infringement theories. EpicRealm's shifting theories use the *same* parts of the Oracle products to satisfy *different* claim requirements. Oracle will describe the products and epicRealm's infringement theories in relation to the following figure prepared by Dr. Clark, which also appears in Oracle's motion for summary judgment of noninfringement.



Oracle's description of the products is based on Dr. Clark's review and explanation of the source code, which defines those structures and functions. A more complete description is in Dr. Clark's accompanying declaration, Clark Opp. Decl. ¶¶ 27-160.

1. Web Cache

Web Cache is a software program designed to maintain a cache, or a local store, of frequently accessed Web pages. Without Web Cache, a request made by a Web client (*e.g.*, a Web browser, such as Internet Explorer) is sent directly to another software program, Oracle HTTP Server ("OHS"), which is a Web server. Web Cache, however, sits in front of OHS, the Web server, and receives the Web client's request for a Web page before OHS does. Clark Opp. Decl. ¶ 28. In one theory, epicRealm contends that Web cache is both the "Web server" and the "dispatcher," even though Web Cache is a single program. In its other two theories, epicRealm ignores Web Cache.

Dr. Clark's review of the source code establishes, however, that Web Cache is a single executable program. There is no "Web server" program in Web Cache separate from a

"dispatcher" program. Clark Opp. Decl. ¶¶ 32, 50-51. EpicRealm's expert, Dr. Finkel, did not review the source code for Web Cache; he has not refuted and cannot refute this fact. A74 [Finkel Dep.] at 82:8-83:4. Thus, while Web Cache contains different features and functions, they are all part of the same program. Clark Opp. Decl. ¶ 32, 50-51.

The Oracle source code also establishes that Web Cache does not distinguish between Web page requests for dynamic content and other requests, such as for static content, and thus cannot "intercept" or "divert" dynamic Web page requests. To the contrary, Web Cache processes all requests in the same way. Clark Opp. Decl. ¶¶ 38-45.

Web Cache is a single multi-threaded program, in which each Web request is delegated to a "fiber" or "thread" within the Web Cache program. Clark Opp. Decl. ¶ 32. Web Cache is further divided into "Front End" fibers that connect to Web clients and perform cache lookups, and "Back End" fibers that connect to the origin server and add content to the cache. *Id*. Neither Web Cache, nor any fiber or thread in Web Cache, is "released" by a "page server" to concurrently process other requests. Clark Opp. Decl. ¶¶ 52-74.

2. IAS

Oracle's Internet Application Server ("IAS") is a large "enterprise" software package that includes numerous features and programs, many of which are unrelated to the Web page technology at issue in this case. IAS includes the OHS Web server program and Oracle Containers For Java ("OC4J") software program, which are involved in servicing Web page requests of all types. The OHS program contains several functions called modules (mods) that are built into a single Web server executable. Clark Opp. Decl. ¶¶ 75-77. In a second theory, epicRealm contends that OHS, which it alleged was a "page server" in the Web Cache theory, is both the "Web server" and the "dispatcher" (but not the "page server"). In this second theory, epicRealm alleges that OC4J is the "page server" and the Database is the "data source."

The OHS program includes an "HTTP Listener" that receives incoming requests and then begins processing them using the modules built into the OHS program. Clark Opp. Decl. ¶ 76. OHS never diverts requests from its normal processing based on whether the requests are for dynamic, as opposed to static, content. OHS processes all requests, static and dynamic, by invoking its modules, one at a time, and then using the appropriate module to process the request. Clark Opp. Decl. ¶¶ 82-83; 90-97.

OHS's determination of which of its modules will process the request is determined by the nature of the Web application that needs to be executed. *Id.* If the request calls for generation of content—static or dynamic—via execution of a Java application, then mod_oc4j is selected. If the request calls for execution of a Perl program, then mod_perl will be selected, and so forth. OHS will always go through the same iterative process of invoking these modules in order to process each request, regardless of whether the requested content is static or dynamic. Requests are never "intercepted" or "diverted" because they call for dynamic, as opposed to static, content. *Id.* As referenced above, these "modules" are not separate software programs; they are all part of the same OHS executable. The module (mod_oc4j) that epicRealm contends is a "dispatcher" is not, therefore, a separate program to which OHS can "route" requests.

When a Web client sends a Web page request (URL) to the computer running OHS, that request is assigned by the computer's operating system to one of the OHS program instances ("processes") running on the Web server machine. In the default configuration, there are 256 OHS processes running at any given time. Clark Opp. Decl. ¶ 81. Once a particular OHS process has received the request, that OHS process begins processing the request. As will be further explained below, neither OHS nor any of these processes is ever "released" by any component epicRealm labels a "page server" to concurrently process other requests.

3. Database

Oracle Database primarily consists of a Relational Database Management System ("RDBMS"). An RDBMS is a complex set of software programs that controls the organization, storage, management, and retrieval of data records in a database (a structured collection of records or data) based on a relational model. The primary role of an RDBMS is to service requests for data stored in the database. Clark Opp. Decl. ¶ 126.

EpicRealm's infringement theory against Database identifies OHS as the "Web server" (but not the "dispatcher" as in the IAS theory). In this third theory, epicRealm's interrogatory responses and expert witness report had identified the JDBC (Java Database Connectivity) and OCI (Oracle Call Interface) libraries as "dispatchers." A95 [ER Response to Interrogatory No. 3] at 2-4 & Tab A; A84 [Finkel Infringement Report], Ex. G, at 63-67. But neither JDBC nor OCI is a separate program from OHS, the alleged "Web server." Clark Opp. Decl. ¶¶ 129-131; 136-138. Rather, JDBC and OCI are software libraries linked into, and part of, both the OHS and OC4J programs. OHS cannot "route" requests to JDBC or OCI, as required by the claims. *Id.*

The "page server" of this third theory is an instance of the Database (referred to as a Real Application Cluster or "RAC"), which epicRealm alleged was the "data source" of the two prior infringement theories. RAC is a database option in which multiple computers, or "nodes," access a common database. Clark Opp. Decl. ¶ 132.

Realizing the problems its first three theories had in identifying a "dispatcher," epicRealm now appears to be creating a new, fourth theory in an (unsuccessful) attempt to solve the problems. In this previously undisclosed theory, OC4J (the "page server" of the second theory) and the JDBC library are labeled a "dispatcher," the Database is a page server, and a shared hard disk is the "data source." *See* ER Br. at 24.

Even if allowed, however, this new theory falls apart because when OHS sends requests

for Java Web pages to an OC4J server, it is that OC4J server that generates the dynamic Web page, not any part of the RAC Database, which simply provides the data records to OC4J. Clark Opp. Decl. ¶¶ 84, 139. Therefore, if OC4J were the "dispatcher," there would be no "page server." In addition, neither JDBC nor OCI transmit requests for Web pages; they transmit database queries. Clark Opp. Decl. ¶¶ 126-131. Neither "examines" or "analyzes" Web page requests, and thus does not meet the "dispatching" requirement of the claims. Clark Opp. Decl. ¶¶ 140-156.[4]

4. Summary Chart of EpicRealm's Shifting Infringement Theories

Oracle has summarized epicRealm's shifting theories in the following chart.

	epicRealm's Web Cache theory	epicRealm's IAS theory	epicRealm's Database theory	epicRealm's new Database theory
"Web server"	Web Cache	OHS	OHS	OHS
"dispatcher"	Web Cache	OHS function mod_oc4j	OCI and JDBC libraries	OC4J, using the JDBC library
"page server"	The IAS product, including the OHS program	OC4J	RAC Database instance	RAC Database instance
"data source"	RAC Database instance	RAC Database instance	a shared hard disk	a shared hard disk

C. EpicRealm's Specious "Dynamic Load Balancing" Arguments

Neither claim 11, nor any asserted claim of the patents-in-suit, refers to "load balancing" in any way. Yet the central focus of epicRealm's case is reading into the claims a requirement of a specific type of "load balancing," based on the use of dynamic information about page servers.

---

[4] EpicRealm's infringement theory also requires that the RAC Database be used in combination with the use of JDBC and OCI connection caching/pooling and runtime load balancing. A95 [ER Response to Interrogatory No. 3] at 2-4 & Tab A; A84 [Finkel Infringement Report], Ex. G, at 63-67. As with "metrics based" load balancing in IAS, few, if any, Oracle customers use JDBC and OCI connection caching/pooling and runtime load balancing, which epicRealm contends are necessary to infringement of claim 11. The use of these features is not, however, relevant to Oracle's noninfringement defenses presented in this brief.

EpicRealm then spins a tale that Oracle obtained the idea from epicRealm's patents, determined that this one type of load balancing was "critical" to Oracle's products, and greatly increased its market share by adding this feature to its products. *See* ER Br. at 6-15. These allegations are wrong and have nothing to do with Oracle's noninfringement defenses presented in this brief.

Oracle has shown in its Opening Claim Construction Brief why epicRealm's attempt to read "dynamic load balancing" into the claims through its lengthy construction of the single term "dispatching" is wrong. That epicRealm did not draft the claims to require load balancing is not surprising. Load balancing, including using dynamic information about the "load" on particular servers to choose the best one to handle a job, was well known and practiced in computer science long before epicRealm applied for the patents-in-suit. *See, e.g.*, A75 [Finkel Dep.] at 180:20-181:5; A108 [Clark Dep.] at 25:13-26:1, 29:11-30:17, 35:14-36:15, 57:14-22. The inventors and their patent attorneys undoubtedly saw no reason to include load balancing in the claims as it would have not provided a sound basis for patentability and would have merely cluttered the claims with additional limitations.

EpicRealm has limited its infringement claims to Oracle products that have the capability of performing what Oracle refers to as "metrics-based load balancing." To epicRealm's great consternation, however, it has learned in discovery that few, if any, Oracle customers use the "metrics-based load balancing" options, preferring instead to use the products without using load balancing or with other load balancing options (*e.g.*, random or "round robin"), which epicRealm contends are not within the scope of the claims. *See* Oracle's Brief in Support of Motion for Summary Judgment of Noninfringement (D.I. 205) at 11, 36.

In the face of the above evidence, epicRealm persists in its attempts to portray Oracle as having taken and greatly benefitted from this nonexistent "dynamic load balancing" invention.

See ER Br. at 8-15. No one at Oracle was aware of the patents-in-suit, however, prior to January 2006, which was well after Oracle added "metrics-based load balancing" to the accused products. *See* Oracle's Brief in Support of Motion for Partial Summary Judgment of No Willful Infringement (D.I. 209) at 10. Nor is there any evidence that customers "demanded" the inclusion of "metrics-based load balancing," as epicRealm represents (ER Br. at 10, citing Nelson Dep.). *See* A80 [Nelson Dep.] at 130:13-18; 133:19-134:2 (referencing "requests" from customers "[f]or better load balancing in general," including metrics-based load balancing, and "more load balancing options").

EpicRealm represents to the Court that Oracle accorded this one load balancing option the "highest possible level of criticality." ER Br. at 15. EpicRealm forgets to mention, however, that the document on which it relies lists all eight load balancing options that may be implemented by OHS as having a high "critical" level. *See* ER Exh. 31 at 99:19-22; 101:14-102:16; ER Exh. 18 at ORCL01050261. By epicRealm's own admission, six of those algorithms do not even come within the scope of the claims. EpicRealm's misplaced fervor to portray "metrics-based load balancing" as the heart and soul of Oracle's products is, at best, bad fiction. Its unfounded implication that Oracle's market share increased from 13% to 20% as a result of adding metrics-based load balancing is even worse. ER. Br. at 10-11. EpicRealm's attempt to base its case on this minor feature of Oracle's products illustrates the contrived nature of epicRealm's infringement arguments.

IV. ARGUMENT

A. The Accused Oracle Products Do Not Infringe Claim 11 Because They Lack The Central Limitations Of The Claim

"If any claim limitation is absent from the accused device, there is no literal infringement as a matter of law." *Bayer AG v. Elan Pharm. Research Corp.*, 212 F.3d 1241, 1247 (Fed. Cir.

2000). As set forth in this brief and in Oracle's own motion for summary judgment of noninfringement, epicRealm's infringement theories fail because each of the accused products lack the "intercepting," "dispatcher," and "releasing" limitations required by all of the asserted claims. EpicRealm's infringement theory against Oracle Database fails for the additional lack of the "dispatching" limitation. The Court should deny epicRealm's motion and grant Oracle's motion.

B. Dynamic Web Page Requests Are Not "Intercepted" Or Diverted At Either Of The Alleged Web Servers (OHS Or Web Cache)

1. EpicRealm's theories against IAS and Database fail because requests are never intercepted or diverted from OHS, the alleged Web server

EpicRealm's argument glosses over the requirement in claim 11 and the other asserted claims that requests for dynamic Web pages are identified and intercepted at the Web server. EpicRealm contends that OHS, the alleged "Web server," meets its proposed construction of "intercepting the handling of a request at a Web server" because OHS "can handle some request on its own while other requests, such as dynamic Web page generation requests, are intercepted" and processed by a "page server." ER Br. at 20-21. EpicRealm is just as vague when explaining how OHS meets the intercepting limitation under Oracle's proposed construction of "receiving a request at the Web server machine and diverting the request before the Web server executable can process the request." EpicRealm merely states that dynamic Web page requests are "diverted . . . before the 'Web server' executable can process the request . . ." and cites as its only support an equally conclusory statement from its expert, Dr. Finkel. Finkel Decl. ¶ 21. Neither epicRealm nor Dr. Finkel dispute the specifics of how OHS operates. These conclusory statements are insufficient to even raise a genuine issue of fact as to infringement, let alone establish infringement as a matter of law. *Dynacore Holdings Corp. v. U.S. Philips Corp.*, 363 F.3d 1263, 1278 (Fed. Cir. 2004) ("It is well settled that an expert's unsupported conclusion on

the ultimate issue of infringement is insufficient to raise a genuine issue of material fact, and that a party may not avoid that rule simply by framing the expert's conclusion as an assertion that a particular critical claim limitation is found in the accused device."); *see also* Oracle's accompanying Objections and Motion to Strike Declaration of Dr. David Finkel ("Obj. to Finkel Decl.").

Neither epicRealm nor Dr. Finkel even attempt to describe where or how intercepting is allegedly taking place. Dr. Finkel admitted that he failed to examine any source code that would permit him to verify epicRealm's "intercepting" theory. Obj. to Finkel Decl. at 1-2 (*citing* A74 [Finkel Dep.] at 82:8-83:4). Dr. Clark, on the other hand, examined the source code for OHS. This source code establishes beyond dispute that OHS treats all requests, whether static or dynamic, the same. There is no "intercepting" or "diverting" of dynamic Web page requests. OHS processes every Web request, whether for dynamic or static content, using the same algorithmic process: Clark Opp. Decl. ¶¶ 81-84; 90-97.

Indeed, the very Oracle documentation that Dr. Finkel cites as supporting his conclusory statements actually supports Dr. Clark's conclusion that no intercepting is present in the IAS product. ER Br., Ex. 19 at 2-2–2-15 at ORCL00370718-370731. Nowhere in the Oracle documentation relied upon by Dr. Finkel, or the source code examined by Dr. Clark, does the processing of a request differ based upon any distinction between dynamic Web page requests and other requests. As described in the Oracle documentation and verified by Dr. Clark's source code analysis, all Web page requests, whether for dynamic Web pages or otherwise, are processed by OHS using the same step-by-step process.

A step-by-step review of how the OHS process functions demonstrates that there is no "intercepting" or "diverting" of dynamic Web page requests from the OHS program, as is

required by the parties' constructions.

First, OHS receives a Web page request from a Web client. This cannot be intercepting because the request, regardless of whether it is for dynamic or static content, is still at the Web server. Clark Opp. Decl. ¶¶ 92-96.

Second, OHS delegates the request to an OHS "process," which is an "instance" or copy of the OHS program. This OHS process then calls each of its modules, one at a time, and compares the Web page request against a list of Web page types that module is designed to accept. Clark Opp. Decl. ¶¶ 81-85; 92-96. This cannot be intercepting because the modules are part of the Web server program and so there is no interception or diversion away from the Web server in any sense. *Id.* Moreover, when OHS calls its various modules, it is not making any distinction based on whether the request is for dynamic content; the call process and sequence is identical in all instances. *Id.* Using the example of a request for a Web page with Java content, when OHS calls its mod_oc4j, OHS will pass that request to OC4J irrespective of whether the request is for a dynamic Web page or not. Further, the step of assigning the request from OHS to its mod_oc4j function cannot be "intercepting" under epicRealm's theory because that step is what epicRealm has identified as "routing to a dispatcher" and this would improperly collapse these two claim limitations, effectively reading "intercepting" out of the claims. *See TIP Sys., LLC v. Phillips & Brooks/Gladwin, Inc.*, 529 F.3d 1364, 1370 (Fed. Cir. 2008) (A claim construction that "conflates construction . . . of separate elements of the claimed invention" is improper.).

Third, OHS identifies the correct module to use and uses that module to satisfy the request. Clark Opp. Decl. ¶¶ 81-85; 92-96. This cannot be intercepting because this is what epicRealm has identified as "dispatching," and would thus vitiate the "intercepting" limitation.

Finally, if OHS calls all of the modules and no appropriate module is identified, it attempts to satisfy the request, regardless of whether it is for static or dynamic content, from data available on the local computer. *Id.* This too cannot be intercepting because the request remains within the OHS program; there still has been no "interception" or "diversion" of the request.

There is no factual dispute that this is how OHS works, and epicRealm cannot escape the fact that OHS lacks the "intercepting" of the patent with its conclusory statements to the contrary. The OHS software simply is not designed to practice the "intercepting" of the asserted claims. EpicRealm's infringement theories against IAS and Database fail for this reason alone.

2. EpicRealm's theory against Web Cache fails because requests are never intercepted or diverted from Web Cache, the alleged Web server

EpicRealm's discussion of how it believes Web Cache satisfies the "intercepting" limitation of claim 11 is even more conclusory than its discussion for the IAS product. Indeed, epicRealm's brief devotes merely a single paragraph to Web Cache's alleged "intercepting" and again cites as its sole support a conclusory paragraph in Dr. Finkel's declaration. Finkel Decl. ¶ 32 (*citing* Report of Dr. Clark, at 10). As with OHS, epicRealm alleges the "intercepting" limitation is met because Web Cache "can handle some requests on its own." ER Br. at 35; Finkel Decl. ¶ 32. Dr. Finkel admits, however, that he never reviewed any Web Cache source code relevant to intercepting. Obj. to Finkel Decl. at 1-2. In fact, Dr. Finkel relies on Dr. Clark's statement of facts to improperly draw the conclusion that "intercepting" exists. But the very page of Dr. Clark's report that Dr. Finkel cites as support actually states that "Web Cache is able to cache both static and dynamic content" and "Cache hits are handled completely by Web Cache, while Web Cache sends cache misses to the origin server for further processing." ER Br., Exh. 7 at 10. As explained by Dr. Clark, Web Cache makes no distinction between "dynamic Web page generation requests that are 'cache misses'" and static Web page requests that are

cache misses. The source code for this functionality within Web Cache, which Dr. Clark reviewed and Dr. Finkel did not, establishes that Web Cache treats all requests, dynamic and static, the same way. Clark Opp. Decl. ¶¶ 38-45. *All cache misses* are sent to OHS, whether static or dynamic. In order to get the patent issued, epicRealm represented to the Patent Office that "merely routing a request from a web server . . . does not involve intercepting." *See* Oracle CC Br. at 12 (quoting A4, at EPIC000497-98).

Dr. Finkel's simple assertion that OHS processes some requests and not others is insufficient to establish "intercepting." EpicRealm and Dr. Finkel do not even attempt to describe where or how intercepting is allegedly taking place. As verified by Dr. Clark's source code analysis, for every request that Web Cache receives, whether for static or dynamic content, the same algorithmic process is followed. Clark Opp. Decl. ¶¶ 38-45.

As with OHS, a step-by-step review of the Web cache process demonstrates that there can be no "intercepting" of dynamic Web page requests.

First, Web Cache receives a Web page request from a Web client. This cannot be intercepting because the request is still at the Web Cache, the alleged "Web server."

Second, Web Cache checks its cache to see if the requested Web page, whether for static or dynamic content, is present. *Id.* If the requested Web page is found within the cache, Web Cache returns the Web page to the client. *Id.* This cannot be intercepting because all of this functionality is contained within the Web Cache program, so there is no intercepting or diverting of the request away from the Web server in any sense.

Third, if the Web page is not found, then Web Cache sends the request to an OHS for further processing. *Id.* This cannot be intercepting because Web Cache's sending of a request to an OHS is what epicRealm has identified as "dispatching," which would vitiate the

"intercepting" limitation.[5]

There is no factual dispute that this is how Web Cache works, and epicRealm cannot escape the fact that Web Cache lacks the "intercepting" of the patent with its conclusory statements to the contrary. Oracle's Web Cache software simply is not designed to practice the "intercepting" of the asserted claims. EpicRealm's infringement theory against Web Cache fails for this reason alone.

C. The Accused Oracle Products Do Not Have A Dispatcher That Performs Dispatching

1. EpicRealm's unsupported assertion that mod_oc4j is the "dispatcher" of IAS fails because mod_oc4j is merely a function of the Web server, OHS

After a request for a dynamic Web page is intercepted at the Web server, claim 11 requires "routing said request from said Web server to a dispatcher." Oracle's construction of "dispatcher" is "a software program for determining which page server should be used to process a dynamic Web page generation request." EpicRealm offers no construction of "dispatcher." *See* Oracle CC Br. at 27-30. Oracle has demonstrated why its construction is correct both in light of the claim language and the specification. *Id.*; Section III.A.2, *supra*.

EpicRealm's infringement theory against IAS identifies OHS as the "Web server" and identifies OHS's mod_oc4j as the "dispatcher." *See* ER Br. at 21-22. There is no dispute, however, that mod_oc4j is merely a function of the OHS program or "executable." Dr. Clark reviewed the Oracle source code and has established that this is true. Clark Opp. Decl. ¶¶ 75-77;

---

[5] As described in Section IV.C.3, *infra*, epicRealm fails to identify any "dispatcher" apart from Web Cache itself. In any event, whatever epicRealm considers the required "routing" of the request from Web Cache as "Web server" to Web Cache as "dispatcher" cannot also be "intercepting," because, again, this would vitiate a claim limitation.

98-101. EpicRealm's expert, Dr. Finkel, did not review the source code and does not dispute this fact. A74 [Finkel Dep.] at 82:8-83:4. Neither he nor epicRealm have made any allegation that that mod_oc4j is anything more than an integrated function within the OHS program.

EpicRealm's theory is wrong because mod_oc4j is not a software program and is merely a function of the alleged "Web server," OHS. Thus mod_oc4j cannot be a "dispatcher" because the Web server would be "routing" the dynamic Web page request to itself, which makes no sense. The claims require "routing" *from* a Web server *to* a dispatcher.

As discussed in Section III.A.2, *supra*, the named inventors of the '554 patent believed that having a separate dispatcher program was important to the "partitioned architecture" they were claiming. Oracle's modern Web servers, unlike prior art Web servers, do not use or need a separate "dispatcher." Clark Opp. Decl. ¶ 100. EpicRealm is thus improperly trying to read the dispatcher requirement out of the claim. Again, the language of claim 11 is instructive: "intercepting said request at said Web server, *routing said request from said Web server to a dispatcher*, and dispatching said request to said page server." EpicRealm's theory seeks to render the italicized language meaningless. According to epicRealm as long as the Web server (OHS) dispatches the request to a page server, even if it skips the step of sending it to a dispatcher, the claim is satisfied. This argument improperly conflates the separate "Web Server" and "dispatcher" elements, and is wrong as a matter of law. *TIP Sys.*, 529 F.3d at 1370. EpicRealm's theory against IAS fails for this reason.

2. EpicRealm's Database theory fails because Oracle Database lacks a dispatcher and does not perform dispatching

a) EpicRealm's theory that OCI and JDBC are dispatchers fails because they are merely part of the Web server, OHS

OCI and JDBC are Application Programming Interfaces ("API") that allow software programs such as OHS and OC4J to communicate with a database. Clark Opp. Decl. ¶¶ 129-

131. The OCI API is used when non-Java requests are sent to the Database. *Id.* These requests do not go from OHS to OC4J, but rather go directly from OHS to the Database. *Id.* The JDBC API is used when Java requests are sent to the database. *Id.* Java requests are processed by OC4J. *Id.* EpicRealm's theory that OCI and JDBC are "dispatchers" fails because OCI and JDBC are not software programs. As Dr. Clark has established by his review of the source code, OCI and JDBC are software libraries that are linked into software programs such as OHS or OC4J. *Id.* EpicRealm and Dr. Finkel do not dispute this fact. Using the example of Microsoft Word again, the Word program links in software libraries such as "printer drivers" that permit Word to send documents to the printer. Clark Opp. Decl. ¶¶ 136-138. The printer driver library is not a software program and cannot run on its own; it merely allows Word to communicate with a printer. The document to be printed is routed from the Word program to the printer. The same is true here: non-Java requests are routed from OHS to a RAC instance of the Database.

There is thus no "routing" from OHS (the alleged Web server) to OCI (an alleged "dispatcher"), as required by the claims. OCI is merely a library linked into the OHS program that allows OHS to communicate with the RAC Database instance. *Id.* Similarly, there is no "routing" from OHS (the alleged Web server) to JDBC (an alleged "dispatcher") because JDBC is merely a library linked into OHS to allow it to communicate with the database instance. *Id.*

EpicRealm does not even attempt to rely on Dr. Finkel to show that OCI or JDBC are "dispatchers," but merely references functions performed by OHS using OCI without any analysis of how OHS could route requests to OCI or how OCI, as opposed to OHS, could route requests to a RAC instance. Instead, epicRealm now attempts to change its theory involving JDBC. The allegations of the new theory are discussed in the next section. Again, epicRealm's theory violates the "all elements" rule by seeking to avoid the "dispatcher" requirement that is

included in claim 11 and all claims. EpicRealm's theory fails as a matter of law.

b) EpicRealm's new theory, if allowed, that OC4J, using JDBC, is the "dispatcher" of Oracle Database fails because OC4J is a "page server"

EpicRealm's interrogatory responses and Dr. Finkel's expert report identified only the JDBC and OCI APIs as "dispatchers" for epicRealm's infringement theory against Database. A84 [Finkel Infringement Report], Ex. G at 63-67. In its brief, epicRealm raises for the first time a new theory that OC4J, using JDBC, is a "dispatcher." ER Br. at 24. Dr. Finkel's declaration does not even purport to support this theory. The reason why epicRealm has proposed this new theory is because it recognizes that JDBC is not a software program at all (but rather a library) and is not a software program to which the alleged Web server, OHS, can route requests, as required by the claims. EpicRealm should not be allowed to raise this new theory, when it was not part of its interrogatory responses or its expert's report. *Inline Connection Corp. v. AOL Time Warner Inc.*, 472 F. Supp. 2d 604, 614 (D. Del. 2007); *Elan Pharma Int'l Ltd. v. Abraxis Bioscience, Inc.*, C.A. No. 06-438 GMS, at 2-4 & n.2 (D. Del. Dec. 6, 2007) (copy provided as A110).

Even if the Court considers epicRealm's new theory, it fails. OC4J cannot be a "dispatcher" because when requests for Web pages are sent from OHS to OC4J—*i.e.*, requests that are for Java content—OC4J itself, not some other part of the system, generates the Web page. Clark Opp. Decl. ¶ 139. In this sense, as epicRealm has alleged in its theory against IAS, OC4J is a "page server." Both parties agree that a "page server" is "page generating software

that generates dynamic Web pages."[6] EpicRealm's theory against Database, however, depends on labeling a *RAC database instance* as the "page server." ER Br. at 25-26. EpicRealm provides three examples of how it contends that a RAC instance can generate a dynamic Web page: "(1) operate XML DB; (2) execute a PL/SQL Web Application; and (3) execute a PL/SQL Server Page." *Id*. at 25 (citations omitted). But none of the requests for these types of alleged dynamic Web pages are sent to OC4J. Clark Opp. Decl. ¶ 139. Only requests for Java content are sent to OC4J and when they are, OC4J, not a RAC instance, generates the dynamic Web page.

When requests are sent to OC4J, therefore, the only "page generating software that generates dynamic Web pages" is OC4J, not an instance of a RAC database. *Id.* Again, epicRealm does not dispute this fact, and cannot offer contrary evidence because its expert did not review the source code. If OC4J, using JDBC, is the alleged "dispatcher," then there is no "page server," and epicRealm's theory fails. Nor can the "dispatcher" be the same program as the "page server." Claim 11 requires "routing said request from said Web server to a dispatcher, and dispatching said request to said page server." An OC4J server cannot, and does not, "dispatch" requests to itself. *Id.*, ¶ 140. EpicRealm's new theory against Database, even if allowed, fails.

c) Oracle Database does not perform dispatching because neither alleged dispatcher analyzes or examines a Web page request to make an informed selection of a page server.

An additional reason that epicRealm's Database theory fails is because neither JDBC nor OCI "analyzes" or "examines" a request for a dynamic Web page to make an informed selection of which page server should process the request, as both parties claim constructions require.

---

[6] Oracle's construction is "page generating software, running on a processor separate from that of the Web server, that generates dynamic Web pages." EpicRealm's construction does not include "running on a processor separate from that of the Web server."

Dr. Clark reviewed the relevant source code sections relating to how JDBC and OCI are involved in allowing OHS and OC4J to communicate with RAC instances. His analysis confirmed that neither JDBC nor OCI can examine a request to make its decision. Clark Opp. Decl. ¶¶ 142-156. JDBC and OCI simply execute the appropriate connection method and a RAC instance is selected, irrespective of the content of the Web page request. *Id.* None of the interfaces for establishing a connection to a RAC instance allow the passing of a dynamic Web page request. *Id.* This is true because RAC instances, which epicRealm alleges are "page servers," are all connected to the same data source. *Id.* When "run time load balancing" is used, JDBC or OCI is merely attempting to choose a RAC instance, which has access to the same data as the other RAC instances. The particular content of the request is not relevant and thus neither JDBC nor OCI has the capability to "analyze" or "examine" it. *Id.*

EpicRealm relies on conclusory allegations in Dr. Finkel's declaration as to how JDBC and OCI "analyze" or "examine" requests. *See* Obj. to Finkel Decl. at 1-3. Dr. Finkel argues that OCI uses "information" on "tagging" and "security attributes and privileges" that "can come from" a Web page request. Finkel Decl. ¶¶ 26-27. He argues that "information" used by JDBC to "'identify connections that are authenticated with the proper user credentials, user name and password' and to 'identify connections that have the necessary striping' can come from a dynamic Web page request." *Id.*, ¶ 28. But Dr. Finkel does not explain what he means by "a Web page request" in this context. Dr. Finkel's conclusions are wrong because the "information" on "tagging," "security attributes and privileges," "user credentials, user name and password," and "connections that have the necessary striping" simply are not part of the Web page request because they are not part of the URL (Uniform Resource Locator) that constitutes the request.

Oracle's claim construction for "request" is "a message that asks for a Web page specified

by a URL." EpicRealm's construction is "a message that asks for a Web page," and thus seeks to create ambiguity as to the message's relation to a URL. The patent leaves no doubt, however, that the request is specified by a URL. *See, e.g.*, col. 4:55-56 (Web client 200 issues a URL request that is processed to determine proper routing."); col. 5:60-62 ("[I]f the URL request requires financial data from data source 408, dispatcher 402 will first examine an information list."); col. 5:64-65 ("Dispatcher 402 will thus route the URL request to Page server 404(3).").

Since the URL request is never passed to JDBC or OCI, they are incapable of analyzing or examining the request to make an informed selection of which RAC instance with which to establish the connection. Clark Opp. Decl. ¶¶ 142-156. That JDBC and OCI use other "information" not contained in the URL request to establish a connection with a RAC instance is irrelevant. This connection ability that is not based on the specific content of the URL does not meet the "dispatching" limitation of the asserted claims as a matter of law and epicRealm's theory fails for this additional reason.

3. EpicRealm's theory that Web Cache is both the "Web server" and the "dispatcher" fails

As Dr. Clark's review of the source code establishes, Web Cache is a single program. Clark Opp. Decl. ¶¶ 32. In its theory against Web Cache, epicRealm identifies that single Web Cache program as both the "Web server" and the "dispatcher." EpicRealm offers no testimony from Dr. Finkel, or any other evidence or explanation, of how Web Cache, as "Web server," can also be the dispatcher, when all claims, including claim 11, require that the Web server route dynamic Web page requests to the dispatcher.

Oracle's modern technology has no need for a dispatcher that is separate from Web Cache. Clark Opp. Decl. ¶¶ 49-51. As discussed above with respect to epicRealm's other infringement theories, its theory against Web Cache fails for lack of a dispatcher.

D. The Accused Oracle Products Cannot Perform The Releasing Limitation

1. EpicRealm's theories against IAS and Database fail because OHS, the accused Web server, is never released by an alleged "page server"

a) EpicRealm's "implicit" releasing theory fails

Claim 11 requires "routing a dynamic Web page generation request from a Web server to a page server, said page server receiving said request and releasing said Web server to process other requests." EpicRealm's construction of the required "releasing" is "freeing." Oracle's construction is "said page server receiving said request and said page server performing an act (separate from merely receiving the request) to free the Web server to process other requests."

EpicRealm argues that OHS, the alleged Web server of its IAS and Database theories, is "released" by an alleged "'page server' . . . processing the request." *See* ER Br. at 19 (identifying OC4J as the "page server") and 26 (identifying a RAC database instance as the "page server"). EpicRealm contends that this satisfies both party's claim constructions, alleging that the page server's processing "frees" the Web server of the "duty" to process the request and that the processing is an act separate from merely receiving the request. The Texas court recently has rejected this same argument: "[T]he Court hereby expressly rejects Plaintiffs' 'implicit releasing theory' where a web server may be released by the page server processing the request routed from the web server." A89 [8/17/08 Order] at 10.

As the Texas court found, epicRealm's argument is wrong as a matter of law because it renders the "releasing" requirement superfluous and effectively reads it out of the claim. Oracle demonstrated in its Opening Claim Construction Brief, at 17-20, that the page server's "receiving" of the request and "releasing" of the Web server must be separate acts. Thus, the mere "offloading" of the request cannot satisfy the "releasing" limitation, as epicRealm argues. *See* ER Br. at 26 ("the 'Web server' offloads the request to the 'page server,' thereby freeing the

'Web server' . . . ."). Nor can the page server's "processing" of the request satisfy the "releasing" limitation. Claim 11 requires the page server to perform both the act of "releasing" and the act of "processing":

> said page server receiving said request and releasing said Web server to process other requests; . . . processing said request, said processing being performed by said page server while said Web server concurrently processes said other requests; . . .

If the "releasing" limitation were satisfied by the page server's "processing" of the request, then the "releasing" limitation would be meaningless; *i.e.*, as long as the page server "processed" the request it "received" from the Web server, "releasing" would *ipso facto* be satisfied. EpicRealm's application of the "releasing" limitation is thus wrong as a matter of law. Oracle CC Br. at 17-20; *TIP Sys.*, 529 F.3d at 1370.[7]

While epicRealm's "implicit releasing" theory is wrong as a matter of claim construction, epicRealm also has failed to provide any evidence that OC4J's processing of a request frees OHS to process other requests. EpicRealm relies solely on Dr. Finkel's declaration, in which he offers only the same conclusory statement found in epicRealm's brief, *i.e.*, that when OC4J processes a Web page request, it releases OHS from the "duty, task or obligation" to process that request. The paragraphs of Dr. Finkel's declaration in which he opines about the "releasing" question are likewise devoid of citation to source code or other factual evidence. Finkel Decl. ¶¶ 17-20. Dr.

---

[7] Oracle's claim construction, which is the Texas court's construction, requires that "releasing" be an act separate from merely "receiving" the request. This is because "receiving" is one of the claimed acts of the page server. Oracle did not perceive that epicRealm would argue that even if "receiving" could not constitute "releasing," other claimed acts of the page server—such as "processing" or "dynamically generating a Web page"—could constitute "releasing." The legal principle is the same, however, and Oracle respectfully requests that this Court hold, as did the Texas court, that neither "receiving" nor any other required acts of the page server can constitute "releasing," *e.g.*, "said page server receiving said request and said page server performing an act (separate from merely receiving or processing the request or dynamically generating a Web page) to free the Web server to process other requests."

Finkel never reviewed the source code that would establish whether Oracle's products practiced the "releasing" limitation. Dr. Clark's examination of the relevant source code shows that OHS and OC4J perform fundamentally different tasks, and that OC4J's processing of a Web page request has no bearing on OHS's processing of other requests, as epicRealm hypothesizes.

Oracle's products simply are not designed to work in the manner epicRealm suggests. EpicRealm's argument is that, by processing a request for a Java-based Web page, the OC4J "page server" frees OHS from doing processing that it would have otherwise had to do itself if OC4J were absent. But this is wrong. If OC4J (or the OHS function mod_oc4j that works in tandem with OC4J) was not installed on the system, OHS would not assume the "duty, task, or obligation" to process those requests. OHS cannot do the type of processing that OC4J performs—OHS would simply go through its algorithm of calling each module, and after none of the modules accepted the request, OHS would attempt in vain to satisfy the request with local data, because OHS does not execute any Java applications on its own. Clark Opp. Decl. ¶¶ 84, 112. OC4J's tasks are fundamentally different than the tasks that OHS was designed to perform, so the mere fact that OC4J performs its processing of a given request does not support epicRealm's assertion that such processing would have been performed by OHS in the absence of OC4J.

Even if OC4J is present and available to process the request, OHS is still never released by this processing. Dr. Clark's analysis of the source code confirms that an instance of the OHS program (which Oracle calls a "process")—the unit of processing responsible for guiding the request through its lifecycle—is not freed to process other requests when it sends the request to OC4J for further processing. Instead, the OHS process waits until the OC4J completes the request or until a time-out error occurs. While there are other OHS processes available to

process requests, those processes were never charged with that first request in the first place, and so logically are never freed by OC4J's processing of it; they remain independent from the OHS process and OC4J that are handling the first request. Clark Opp. Decl. ¶¶ 107-112.

Dr. Finkel in his declaration draws an analogy of an employer delegating her tasks to an employee, and by virtue of having the employee perform the tasks, the employer is "released" to do other things. This analogy is disproven by the source code that Dr. Finkel did not review. The more accurate analogy would be that there are many employers (processes), and when one employer gives a task to an employee, the employer waits until the employee completes the task. Further, in the absence of the employee, the employer could not perform this type of task, and the task would remain incomplete. The employer in Dr. Finkel's flawed analogy is never freed by her employee to perform other tasks. *Id.*

b) EpicRealm's "explicit" releasing theory fails

(1) OC4J, the alleged page server, does not send the TCP ACK message; the operating system sends ACK messages

EpicRealm argues in the alternative that an ACK message sent by the TCP layer of an operating system satisfies the "releasing" limitation of claim 11. Oracle does not supply operating systems and the operating system is not part of the "page server" under either party's construction. Oracle's construction of "page server" is "page generating software, running on a processor separate from that of the Web server, that generates dynamic Web pages." EpicRealm's construction is "page generating software that generates dynamic Web pages."

The Texas court recently held that a "page server" could include an operating system, but that it would have to be a separate operating system from that of the "Web server." The court found the separation necessary to maintain the claimed requirement of a "partitioned architecture," because otherwise the "Web server" could not "route" requests to the "page

server," as required by the claims. A88 [8/7/08 Order] at 5-7. Without such a separation of operating systems, "'the Web server would effectively be 'routing' requests from itself to itself.'" *Id.* at 7 n.3. The problem with the Texas court's analysis that the page server can include an operating system, however, is its failure to take into account (apparently because it was not argued) that there is a required intermediate software program—the dispatcher—to which the Web server must "route" requests and which must "dispatch" those requests to the page server. *See* Section IV.C, *supra*.

The '554 patent could not possibly contemplate ACK messages sent by the operating system of a "page server" to constitute "releasing" of the "Web server." Those ACK messages are being sent because the *dispatcher* software program, not the Web server software program, dispatches a request to the page server. Clark Opp. Decl. ¶¶ 114-125. The dispatcher's receipt of the Web page request from the Web server would cause the *operating system on which the dispatcher* is running to send an ACK message to the Web server, regardless of whether the request is ever successfully dispatched to the "page server." *Id.* ¶ 121. Thus, if the sending of an ACK message could be considered "releasing" of the Web server, it would be the operating system of the dispatcher, not the operating system of the page server, that would be performing the "releasing." This is *not* what the claims require.

The '554 patent explicitly teaches against epicRealm's allegation that an ACK message generated by the "page server's" operating system in response to receipt of a request for a dynamic Web page is in any way "releasing" the Web server. As discussed in Section III.A.2, *supra*, the patent states that the preferred embodiment of the claimed invention places the dispatcher software program on a machine separate from that of the Web server. Col. 5:8-36. The separate operating system of this stand-alone dispatcher would send ACK messages as a

result of the dispatcher's receipt of the request from the Web server. Clark Opp. Decl. ¶ 121. A "page server's" receipt of the request from the dispatcher, if it happens at all, would send an ACK to the dispatcher's operating system, not to the Web server's. *Id.* The preferred embodiment would thus not be covered by the claims because the ACK message from the page server's operating system to the dispatcher's operating system would not "release" the Web server or its operating system. *Id.* "A claim interpretation that reads out a preferred embodiment 'is rarely, if ever, correct and would require highly persuasive evidentiary support.'" *Amgen Inc. v. Hoechst Marion Roussel, Inc.*, 314 F.3d 1313, 1349 (Fed. Cir. 2003) (quoting *Vitronics Corp. v. Conceptronic, Inc.*, 90 F.3d 1576, 1583 (Fed. Cir. 1996)). Nor would construing "page server" to include the operating system cover the alternative embodiment in which the dispatcher program resides on the same machine as the Web server. Col. 5:20-36; Clark Opp. Decl. ¶ 122. The TCP message still would be sent by the "page server" operating system as a result of the receipt of request from the dispatcher software program, not from the Web server program (executable). Clark Opp. Decl. ¶ 122.

(2) Even if the operating system is part of the "page server," the sending of any ACK message does not release OHS to process other requests

Even if the computer operating system is considered part of the "page server," epicRealm's TCP ACK theory fails. First, the "page server's" ACK message is going to the "dispatcher," not to the "Web server," as explained above. Second, in addition to this crucial distinction, Dr. Clark's analysis of the source code establishes that the receipt of any number of ACK messages at the Web server's operating system has no effect on how OHS processes requests. OHS contains 256 OHS processes in its default setting. The receipt of an ACK from an OC4J does not free the OHS process affiliated with that request received by OC4J. The OHS process will continue to wait in an idle state until OC4J returns the requested Web page content,

or until a time-out error occurs. Clark Opp. Decl. ¶¶ 123-124. Nor will the ACK free the other 255 OHS processes in any way—they are independent from that OC4J's actions, and will continue to do whatever they are doing. *Id.* EpicRealm and its expert, Dr. Finkel, point to no evidence to support their theory that an ACK somehow releases any OHS process to process other requests, as required by claim 11 and the other asserted claims.

(3) Even if "freeing of resources" hypothetically could satisfy the releasing limitation, epicRealm's theory fails

EpicRealm realizes that no OHS process is ever "released" to process other requests by an ACK message. EpicRealm attempts to sidestep the issue by asserting that the "freeing of resources" is sufficient to satisfy the limitation of "releasing said Web server to process other requests." EpicRealm states that when a Web server receives an ACK message, the ACK enables the operating system to free computer memory that was allocated to the Web server's TCP buffer, and this freed memory permits OHS to process other requests. ER Br. at 20. This theory fails for at least two reasons. First, the "freeing of resources" argument suffers from the same problem discussed above. The patent cannot possibly contemplate that an ACK message from the page server's operating system "releases" the Web server by "freeing resources" on the Web server. The "memory freed," if any, by the page server operating system would be on the dispatcher's operating system, not on the Web server's operating system. Clark Opp. Decl. ¶ 121.

Second, even ignoring this critical problem, epicRealm has not even attempted to show how TCP buffer memory released on the Web server's operating system allows the Web server to "process other requests" in any way, either by processing additional requests or by processing those it is already handling "more efficiently." Each OHS process has its own TCP buffer. If that TCP buffer receives an ACK, the "freed memory" need not be allocated to any other OHS process on the machine. Rather, that "freed memory" may be allocated to any number of other

programs on the machine, such as firewalls, system management programs, and other networking programs. Clark Opp. Decl. ¶ 123. Dr. Finkel has done absolutely no analysis to establish that the "freeing" of memory in a TCP buffer has any effect whatsoever on the Web server's processing of other requests. EpicRealm's contrived theory based on the Web server's alleged receipt of ACK messages (which do not come from a "page server") fails.

2. EpicRealm's theory against Web Cache fails because Web Cache, the accused Web server, is never released by an alleged "page server"

a) EpicRealm's "implicit" theory fails

EpicRealm advances the same "implicit" releasing theory when it contends that Web Cache is the "Web server" as it does when it contends that OHS is the "Web server." *See* ER Br. at 34-35. As the Texas court found, this theory is wrong as a matter of law for the same reasons discussed in Section IV.D.1(a), *supra*; that is, it improperly reads the "releasing" limitation out of the claim by equating "releasing" with "processing," which is a separate and distinct claim limitation.

b) EpicRealm's "explicit" theory fails

EpicRealm's TCP ACK theory fails for Web Cache as it does with OHS. First, epicRealm has not even identified a "page server" (page generating software) for this theory other than "IAS." But IAS is the name of an Oracle product that is comprised of several software programs; it is not a software program itself. The page generating software is either OHS or OC4J. Second, neither OHS not OC4J sends ACK messages. Only operating systems send ACK messages. Clark Opp. Decl. ¶¶ 59-69. Third, even if an operating system or systems is considered to be part of the undefined "page server," an ACK message from OC4J's operating system cannot free Web Cache because it is sent to OHS's operating system, not Web Cache's operating system. *Id.* An ACK message sent by OHS's operating system to Web Cache's

operating system could not be considered an ACK to a "Web server" because this would ignore the requirement that the "page server" communicate with the "dispatcher," not the "Web server." *Id.* Fourth, ACK messages sent from OHS to Web Cache do not release Web Cache to process other requests. As Dr. Clark's undisputed source code analysis demonstrates, the Web Cache fibers wait when they send requests to OHS—they are not freed to process other requests. Clark Opp. Decl. ¶¶ 59-69. Finally, even if "releasing resources" such as operating system memory were considered to be a way that a Web server could be released, epicRealm has failed to show (1) that an ACK from a "page server" is causing Web Cache's operating system memory to be released or (2) that any such memory is being allocated to the Web Cache, as opposed to other programs on the Web Cache machine. *Id.* EpicRealm's TCP ACK theory therefore fails for all of these reasons.

E. Oracle Does Not Make Or Sell A Machine Readable Medium That Meets The Limitations Of Claim 11

Even if epicRealm's infringement theories were correct, the software Oracle sells is insufficient to cause a computer system to perform the required steps of claim 11. Oracle's software does not send or receive ACK messages; only operating systems, which Oracle does not supply, send or receive ACK messages. Clark Opp. Decl. ¶¶ 67-69. In *Padcom, Inc. v. Netmotion Wireless, Inc.*, No. Civ. 03-983-SLR, 2006 WL 416865 (D. Del. Feb. 22, 2006), this Court held that "a software product can infringe when it interacts with an operating system on a computer, *so long as the software, without the operating system, contains all the limitations set out in the claim*." *Id.* at *5 (emphasis added). Here, it is undisputed that the Oracle software is incapable of sending the ACK message required by epicRealm's "releasing" theory. In *Padcom*, this Court found that the accused software infringed a "computer readable medium claim" because "even if the operating system supplies the information to the software, the software, if it

ascertains the availability of the networks, directly infringes." *Id.* at *6. Here, Oracle software does not send or receive the ACK messages and therefore does not directly infringe the claim. Clark Opp. Decl. ¶¶ 67-69. Indeed, to prove direct infringement under its theories, epicRealm would have to prove specific instances of customers setting up "Web servers" and "page servers" on separate operating systems. *See* A88 [8/7/08 Order] at 5-7. EpicRealm has not offered any such proof. Its motion for partial summary judgment of literal infringement should, therefore, be denied.

V. CONCLUSION

For the foregoing reasons, the Court should deny epicRealm's motion for partial summary judgment of infringement of claim 11 of the '554 patent, and should grant Oracle's motion for summary judgment of noninfringement of all the asserted claims of the patents-in-suit.

MORRIS, NICHOLS, ARSHT & TUNNELL LLP

/s/ James W. Parrett, Jr.

Mary B. Graham (#2256)
James W. Parrett, Jr. (#4292)
1201 N. Market Street
P.O. Box 1347
Wilmington, DE 19899-1347
(302) 658-9200
mgraham@mnat.com
jparrett@mnat.com
Attorneys for Oracle Corporation and Oracle U.S.A., Inc.

OF COUNSEL:

James G. Gilliland
Theodore T. Herhold
Joseph A. Greco
Robert J. Artuz
Eric M. Hutchins
TOWNSEND AND TOWNSEND AND CREW LLP
379 Lytton Avenue
Palo Alto, CA 94301
(650) 326-2400

Dorian Daley
Peggy E. Bruggman
Matthew M. Sarboraria
ORACLE CORPORATION
ORACLE U.S.A., INC.
500 Oracle Parkway
Redwood Shores, CA 94065

August 21, 2008
61454410 v2

CERTIFICATE OF SERVICE

I hereby certify that on August 21, 2008, I caused the foregoing to be electronically filed with the Clerk of the Court using CM/ECF which will send electronic notification of such filing to the following:

Richard L. Horwitz
David Ellis Moore
POTTER ANDERSON & CORROON, LLP

Additionally, I hereby certify that true and correct copies of the foregoing were caused to be served on August 21, 2008 upon the following individuals in the manner indicated:

BY E-MAIL

Richard L. Horwitz
David Ellis Moore
POTTER ANDERSON & CORROON, LLP
1313 N. Market St., Hercules Plaza, 6th Floor
P.O. Box 951
Wilmington, DE 19899-0951

rhorwitz@potteranderson.com
dmoore@potteranderson.com

BY E-MAIL

George S. Bosy
JENNER & BLOCK
330 N. Wabash Avenue
Chicago, IL 60611-7603

gbosy@jenner.com

/s/ James W. Parrett, Jr.
James W. Parrett, Jr. (#4292)

61454410 v2